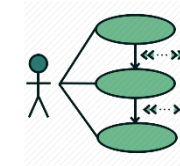


Use Case User Story Unified Modeling Language Quality Attributes

Use Case Name	Name of the use case
Actors	[An actor is a person or other entity external to the system being specified who interacts with the system and performs use cases to accomplish tasks]
Preconditions	[Activities that must take place, or any conditions that must be true, before the use case can be started]
Normal Flow	[User actions and system responses that will take place during execution of the use case under normal, expected conditions]
Postconditions	[State of the system at the conclusion of the use case execution with a normal flow (nominal)]
Alternative flows and exceptions	[Major alternative flows or exceptions that may occur in the flow of event]
Non functional requirements	[All non-functional requirement: e.g., dependability (safety, reliability, etc.), performance, ergonomic]



CENG 323 – Project Management

9 November 2022, @IZTECH

Image credits:

Use case: https://www.researchgate.net/publication/290219988_Hazard_analysis_of_human-robot_interactions_with_HAZOP-UML/figures?lo=1

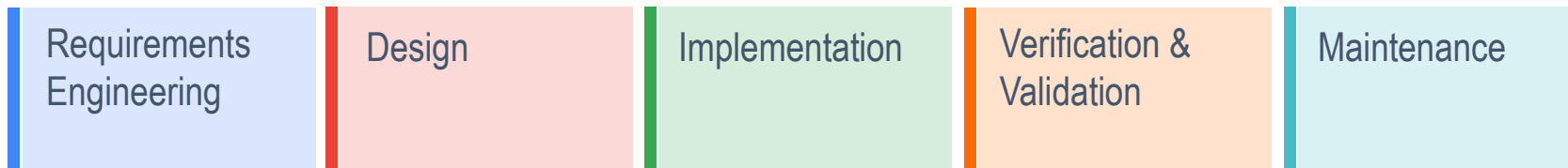
https://www.iconfinder.com/icons/1483069/case_use_business_scenarios_interaction_portfolio_uml_use-case_icon

User story: <https://agiledigest.com/agile-digest-tutorial-2/user-story/>

UML: <https://www.omg.org/uml-directory/>

Recall from
Week 1

Main Activities in Software Engineering

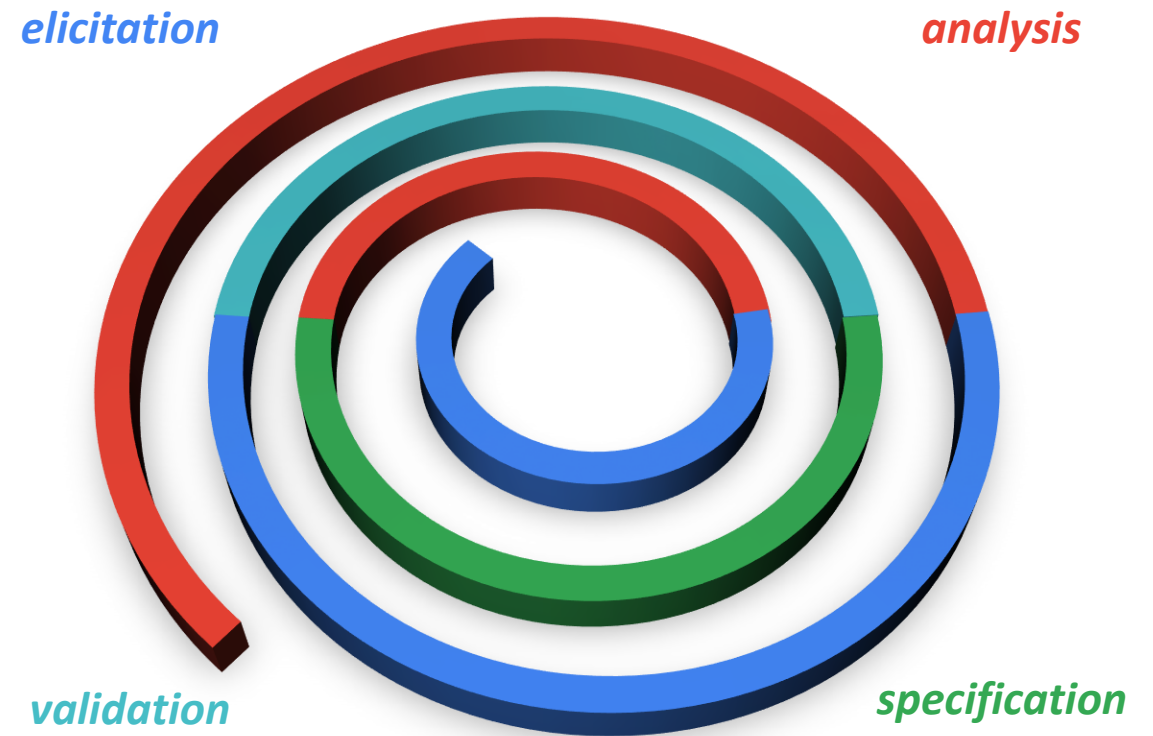


Recall from
Week 1

Requirements Engineering

establish the services that the customer requires from the software system

WHAT SHOULD THE SOFTWARE SYSTEM DO?



From Ideas to Software Systems



IDEA

Business Requirements
Objectives – WHY?
Project Charter

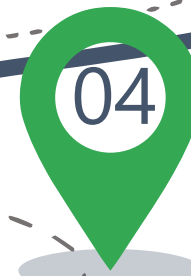


Detailed Requirements
WHAT?
Software Requirements
Specification (SRS)



Problem Analysis
AS-IS Business Processes
TO-BE Business Processes

User Requirements
WHAT?
Use Cases
User Stories

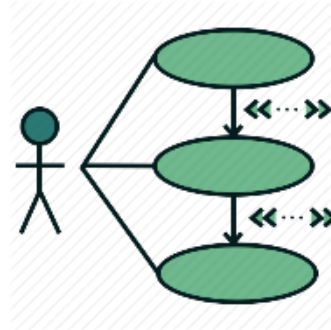


SOFTWARE SYSTEM

Solution Design &
Implementation
HOW?

Use Case

Use Case Name		[Name of the use case]
Actors		[An actor is a person or other entity external to the system being specified who interacts with the system and performs use cases to accomplish tasks]
Preconditions		[Activities that must take place, or any conditions that must be true, before the use case can be started]
Normal Flow	Description	[User actions and system responses that will take place during execution of the use case under normal, expected conditions.]
	Postconditions	[State of the system at the conclusion of the use case execution with a normal flow (nominal)]
Alternative flows and exceptions		[Major alternative flows or exceptions that may occur in the flow of event]
Non functional requirements		[All non-functional requirement: e.g., dependability (safety, reliability, etc.), performance, ergonomic]



Sample Case: Point of Sale (POS) System

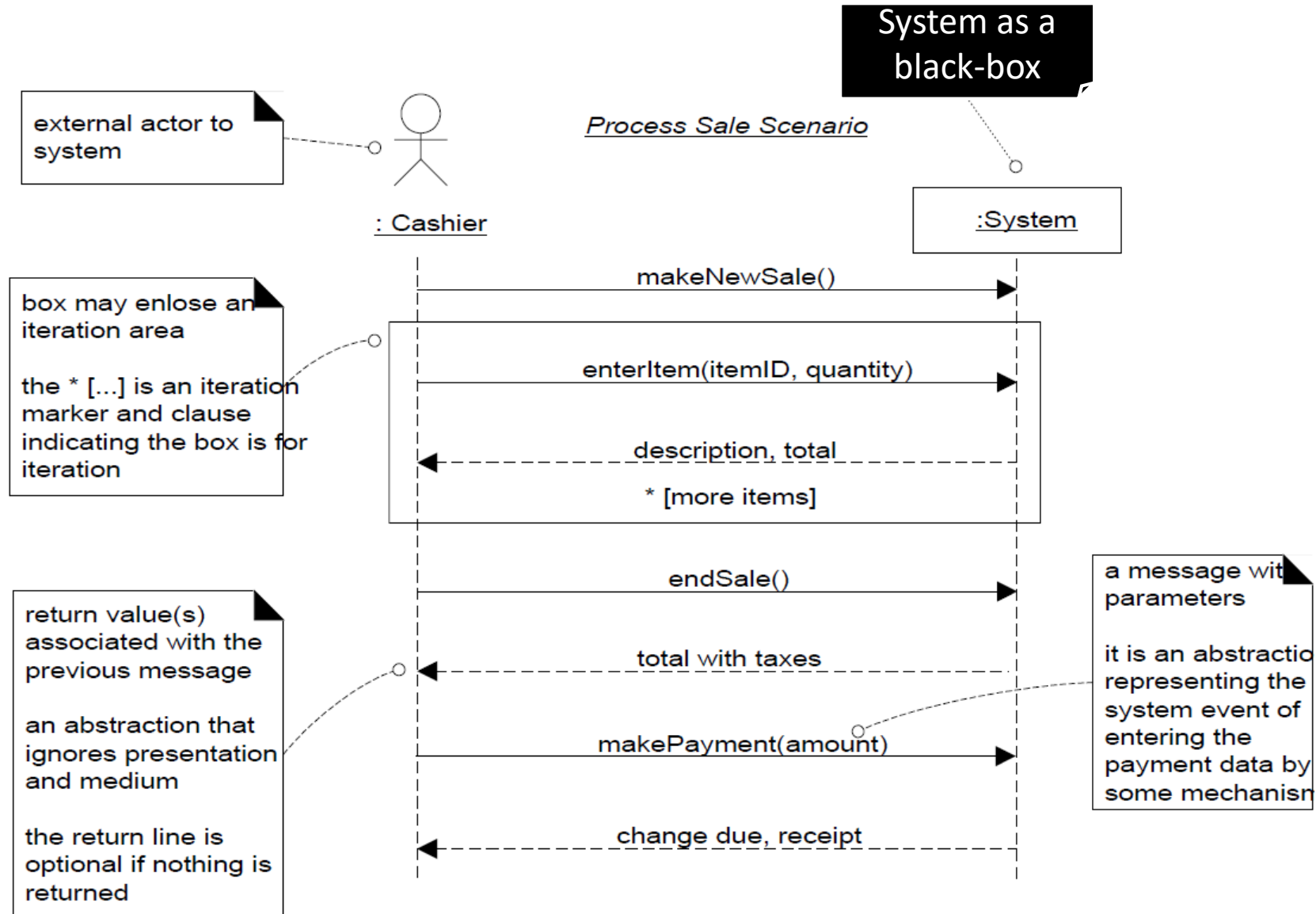


Image credits:

<https://www.pm365.ga/ProductDetail.aspx?iid=46336781&pr=>

<http://www.interstatecashregister.com/grocery/>

Sequence Diagram for “Process Sale” Use Case



What is a “Use Case”?

- a **collection** of related **success and failure scenarios** that describe an **actor** using a system to support a **goal**
- describes a sequence of **interactions** between a system and an external actor that results in the actor being able to **achieve some outcome of value**
- is not a diagram, it is a text

ID and Name:	UC-1 Process Sale		
Primary Actor:	Cashier	Secondary Actors:	Customer, Company, Manager, Government Tax Agencies, Payment Authorization Service
Preconditions:	PRE-1: Cashier is identified and authenticated.		
Postconditions:	POST-1: Sale is saved. POST-2: Tax is correctly calculated. POST-3: Accounting and Inventory are updated. POST-4: Receipt is generated. POST-5: Payment authorization approvals are recorded.		
Normal Flow (Basic Flow, Main Success Scenario):	1. Customer arrives at POS checkout with goods and/or services to purchase. 2. Cashier starts a new sale. 3. Cashier enters item identifier. 4. System records sale line item and presents item description, price, and running total. Price calculated from a set of price rules. Cashier repeats steps 3-4 until indicates done. 5. System presents total with taxes calculated. 6. Cashier tells Customer the total, and asks for payment. 7. Customer pays and System handles payment. 8. System logs completed sale and sends sale and payment information to the external Accounting system (for accounting and commissions) and Inventory system (to update inventory). 9. System presents receipt. 10. Customer leaves with receipt and goods (if any).		
Alternative Flows (Extensions, Exceptions):	3a. Invalid item ID (not found in system): 1. System signals error and rejects entry. 2. Cashier responds to the error: 2a. There is a human-readable item ID (e.g., a numeric UPC): 1. Cashier manually enters the item ID. 2. System displays description and price. 2a. Invalid item ID: System signals error. Cashier tries alternate method. 2b. There is no item ID, but there is a price on the tag: 1. Cashier asks Manager to perform an override operation. 2. Managers performs override. 3. Cashier indicates manual price entry, enters price, and requests standard taxation for this amount (because there is no product information, the tax engine can't otherwise deduce how to tax it)		
Special Requirements:	– Touch screen UI on a large flat panel monitor. Text must be visible from 1 meter. – Credit authorization response within 30 seconds 90% of the time.		
Priority:	High		
Frequency of Use:	Could be nearly continuous.		

Scenario

- a specific sequence of actions and interactions between actors and the system
- also called a *use case instance*



Sample Use Cases from Various Applications

Application	Sample use case	
Point of Sale (POS) system	<ul style="list-style-type: none">• Process Sale• Handle Returns	<ul style="list-style-type: none">• Manage Security• Manage Users
Chemical tracking system	<ul style="list-style-type: none">• Request a Chemical• Print Material Safety Data Sheet• Change a Chemical Request	<ul style="list-style-type: none">• Check Status of an Order• Generate Quarterly Chemical-Usage Reports
Airport check-in kiosk	<ul style="list-style-type: none">• Check in for a Flight• Print Boarding Passes• Change Seats	<ul style="list-style-type: none">• Check Luggage• Purchase an Upgrade
Accounting system	<ul style="list-style-type: none">• Create an Invoice• Reconcile an Account Statement• Enter a Credit Card Transaction	<ul style="list-style-type: none">• Print Tax Forms for Vendors• Search for a Specific Transaction
Online bookstore	<ul style="list-style-type: none">• Update Customer Profile• Search for an Item• Buy an Item	<ul style="list-style-type: none">• Track a Shipped Package• Cancel an Unshipped Order

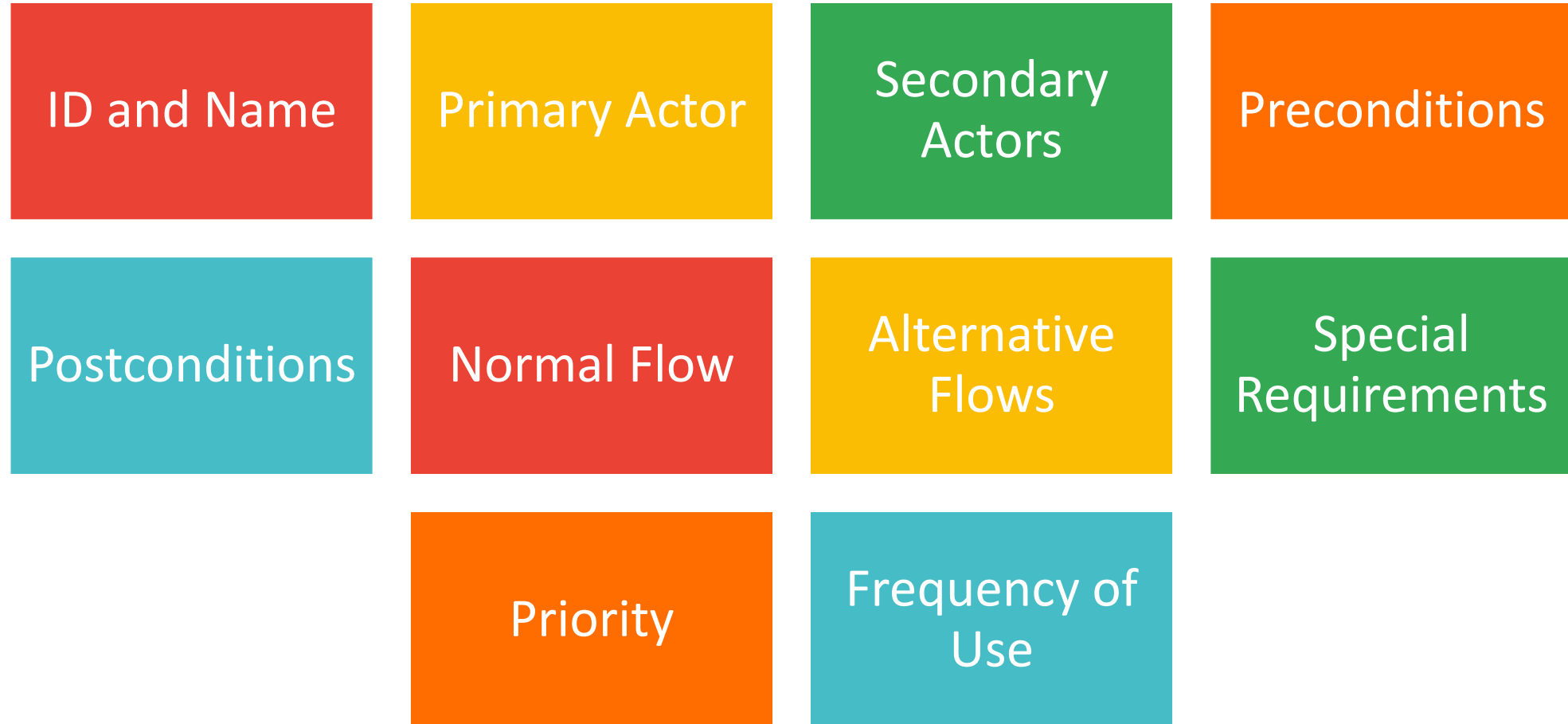


- is a person (or sometimes another software system or a hardware device) that interacts with the system to perform a use case
- Some questions for identifying actors:
 - Who (or what) is notified when something occurs within the system?
 - Who (or what) provides information or services to the system?
 - Who (or what) helps the system respond to and complete a task?

User vs. Actor

- **User** has a collection of hats available, each labeled with the name of an **actor** that the system will recognize as participating in certain use cases.
- **Users** are actual people (or systems); **actors** are abstractions.

Elements of Use Case



Unique ID and Name

A unique identifier and a concise name that states the user goal

ID and Name:	UC-1 Process Sale
---------------------	-------------------

Naming use cases:

- reflect the actor's goal from the actor's perspective
- describe a valuable transaction
- be general enough to cover related scenarios
- use active verb + object
 - "Process Sale", not "Sale process"
 - use strong verbs and specific nouns

Good Examples

Reserve Rental Car
Print Invoice
Check Flight Status

Not So Good Examples

Enter PIN
Submit Form 37

Primary Actor

The principal actor that calls upon system services to fulfill a goal.

Primary Actor:	Cashier
-----------------------	---------

Secondary Actors

A use case, as the contract for behavior, captures *all and only* the behaviors related to satisfying the stakeholders' interests [Alistair Cockburn].

Secondary Actors:	Customer, Company, Manager, Government Tax Agencies, Payment Authorization Service
--------------------------	--

Example:

- Customer: Wants purchase and fast service with minimal effort. Wants easily visible display of entered items and prices. Wants proof of purchase to support returns.
- Government Tax Agencies: Want to collect tax from every sale. May be multiple agencies, such as national, state, and county.

Preconditions

Zero or more preconditions that must be satisfied before the use case can begin

- state what *must always* be true before a scenario is begun in the use case
- are *not* tested within the use case
- implies typically a scenario of another use case

Preconditions:	PRE-1: Cashier is identified and authenticated.
-----------------------	---

Postconditions

One or more postconditions that describe the state of the system after the use case is successfully completed

- state what must be true on successful completion of the use case
- should meet the needs of all stakeholders

Postconditions:	POST-1: Sale is saved. POST-2: Tax is correctly calculated. POST-3: Accounting and Inventory are updated. POST-4: Receipt is generated. POST-5: Payment authorization approvals are recorded.
------------------------	---

Postconditions

- describe the state of the system after the use case executed successfully
- can describe:
 - **Something observable** to the user (the system displayed an account balance).
 - **Physical outcomes** (the ATM has dispensed money and printed a receipt).
 - **Internal system state changes** (the account has been debited by the amount of a cash withdrawal, plus any transaction fees).

Normal Flow

A numbered list of steps that shows the sequence of interactions between the actor and the system—a dialog—that leads from the preconditions to the postconditions

also called the main flow, basic flow, normal course, primary scenario, main success scenario, sunny-day scenario, and happy path

Normal Flow (Basic Flow, Main Success Scenario):

1. Customer arrives at POS checkout with goods and/or services to purchase.
2. Cashier starts a new sale.
3. Cashier enters item identifier.
4. System records sale line item and presents item description, price, and running total. Price calculated from a set of price rules.
Cashier repeats steps 3-4 until indicates done.
5. System presents total with taxes calculated.
6. Cashier tells Customer the total, and asks for payment.
7. Customer pays and System handles payment.
8. System logs completed sale and sends sale and payment information to the external Accounting system (for accounting and commissions) and Inventory system (to update inventory).
9. System presents receipt.
10. Customer leaves with receipt and goods (if any).

Normal Flow

- describes a typical success path that satisfies the interests of the stakeholders
- records the steps, of which there are three kinds:
 1. An interaction between actors.
 2. A validation (usually by the system).
 3. A state change by the system (for example, recording or modifying something).

Alternative Flows

deliver the same business outcome (sometimes with variations) as the normal flow but represent less common or lower-priority variations in the specifics of the task

Exceptions: describe anticipated error conditions that could occur during execution of the use case and how they are to be handled

Alternative Flows (Extensions, Exceptions):

- 3a. Invalid item ID (not found in system):
 - 1. System signals error and rejects entry.
 - 2. Cashier responds to the error:
 - 2a. There is a human-readable item ID (e.g., a numeric UPC):
 - 1. Cashier manually enters the item ID.
 - 2. System displays description and price.
 - 2a. Invalid item ID: System signals error. Cashier tries alternate method.
 - 2b. There is no item ID, but there is a price on the tag:
 - 1. Cashier asks Manager to perform an override operation.
 - 2. Managers performs override.
 - 3. Cashier indicates manual price entry, enters price, and requests standard taxation for this amount (because there is no product information, the tax engine can't otherwise deduce how to tax it)

Special Requirements

include qualities such as performance, reliability, and usability, and design constraints (often in I/O devices)

Example:

- Touch screen UI on a large flat panel monitor. Text must be visible from 1 meter.
- Credit authorization response within 30 seconds 90% of the time.

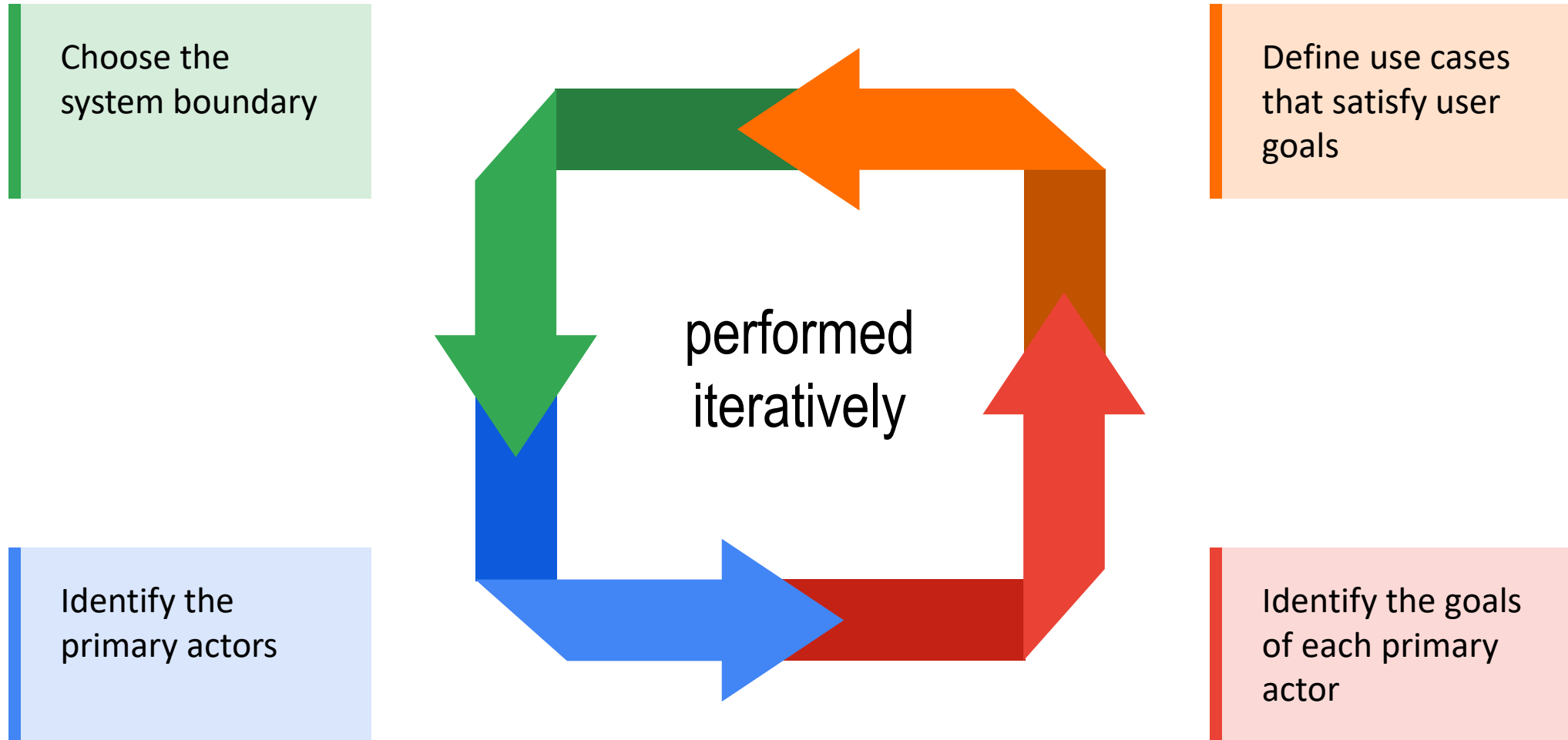
Guidelines for Writing Good Use Cases

- Write Black-Box Use Cases
 - **what** the system must do without deciding **how** it will do it (the design)

Black-box style	Not
The system records the sale.	The system writes the sale to a database. ...or (even worse): The system generates a SQL INSERT statement for the sale...

- Take an Actor and Actor-Goal Perspective
 - write requirements focusing on the users or actors of a system, asking about their goals and typical situations
 - focus on understanding what the actor considers a valuable result
- Essential Style Writing
 - Write use cases in an essential style; **keep the user interface out** and **focus on actor intent**

Guideline: How to Find Use Cases



Step 1: Choose the System Boundary

- system under design
- if not clear, define what is outside (the external primary and supporting actors)

Steps 2 and 3: Find Primary Actors and Goals

Questions to help find actors and goals

- Who starts and stops the system?
- Who does system administration?
- Who does user and security management?
- Is “time” an actor because the system does something in response to a time event?
- Is there a monitoring process that restarts the system if it fails?
- Who evaluates system activity or performance?
- How are software updates handled? Push or pull update?
- Who evaluates logs? Are they remotely retrieved?
- In addition to human primary actors, are there any external software or robotic systems that call upon services of the system?
- Who gets notified when there are errors or failures?

Steps 2 and 3: Find Primary Actors and Goals

Organize the Actors and Goals

1. As you discover the results, draw them in a use case diagram, naming the goals as use cases.
2. Write an actor-goal list first, review and refine it, and then draw the use case diagram.

Steps 2 and 3: Find Primary Actors and Goals

An actor-goal list

Actor	Goal	Actor	Goal
Cashier	process sales process rentals handle returns cash in cash out ...	System Administrator	add users modify users delete users manage security manage system tables ...
Manager	start up shut down ...	Sales Activity System	analyze sales and performance data

Step 4: Define Use Cases

- define one use case for each user goal
 - Goal: process a sale; Use Case: *Process Sale*.
- collapse CRUD (create, retrieve, update, delete) separate goals into one CRUD use case
 - *Manage Users*

What use cases describe and when to use them

Use cases describe

- user goals
- the user's view of the system
- a set of task-related activities

Use cases do NOT describe

- user interface designs
- technology solutions
- application architecture

Use cases work well for

- business automation projects
- websites
- devices with which users must interact

Use cases are NOT as valuable for

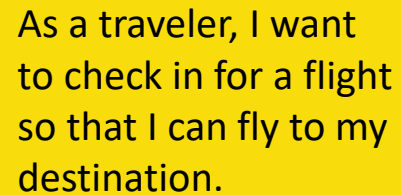
- event-driven real-time systems
- computationally-intensive systems
- business analytics/reporting systems

User Story

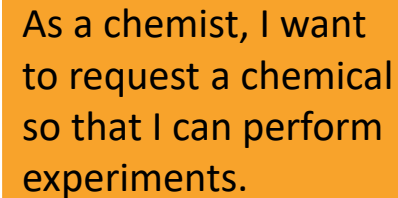


User Story

- is a convenient format for expressing the desired business value for many types of product backlog items (PBI), especially features
- describes functionality that will be valuable to a user of a system or software

A yellow rectangular sticky note with a folded bottom-right corner, containing a user story text.

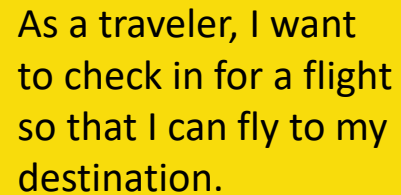
As a traveler, I want to check in for a flight so that I can fly to my destination.

An orange rectangular sticky note with a folded bottom-right corner, containing a user story text.

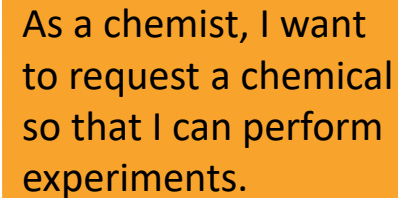
As a chemist, I want to request a chemical so that I can perform experiments.

Motivation for User Story Usage

- a **lightweight** approach compatible with core agile principles
- represent requirements in a **short format** for **detailing** further when and if needed

A yellow rectangular sticky note with a folded bottom-right corner, containing a user story.

As a traveler, I want
to check in for a flight
so that I can fly to my
destination.

An orange rectangular sticky note with a folded bottom-right corner, containing a user story.

As a chemist, I want
to request a chemical
so that I can perform
experiments.

3 Aspects (3 Cs) of User Story



Card



Conversation



Confirmation

Card

- can be a 3 x 5 inch card or sticky notes or ...
- promotes conciseness

Connextra			A Connextra Story Card		
Perspective	Title	Reserved for priority			
	WRITING GOOD STORIES				
Reason	As a Connextra employee - I want to know how to write good stories so that I can submit cards to the planning game that are clear and will be accepted in the next iteration.		Requirements		
Author	Date	Reserved for estimate			
Tim	8/Nov/01				

User Story Title

As a <user role>, I want to <goal> so that <benefit>.

Conversation



- aims to detail a requirement
- takes place among development team, product owner, and stakeholders
- may take place many times (initial, refining, estimating, sprint planning, designing, building, testing)
- is frequently supplemented with documents



Confirmation

- are acceptance criteria that clarify the desired behavior
- used by development team to better understand what to build and test
- used by the product owner to confirm that the user story has been implemented to his satisfaction
- can be specified on the back of the card

3 Cs - Example

Upload File
As a wiki user, I want
to upload a file to the
wiki so that I can
share it with my
colleagues.

a User Story on a Card

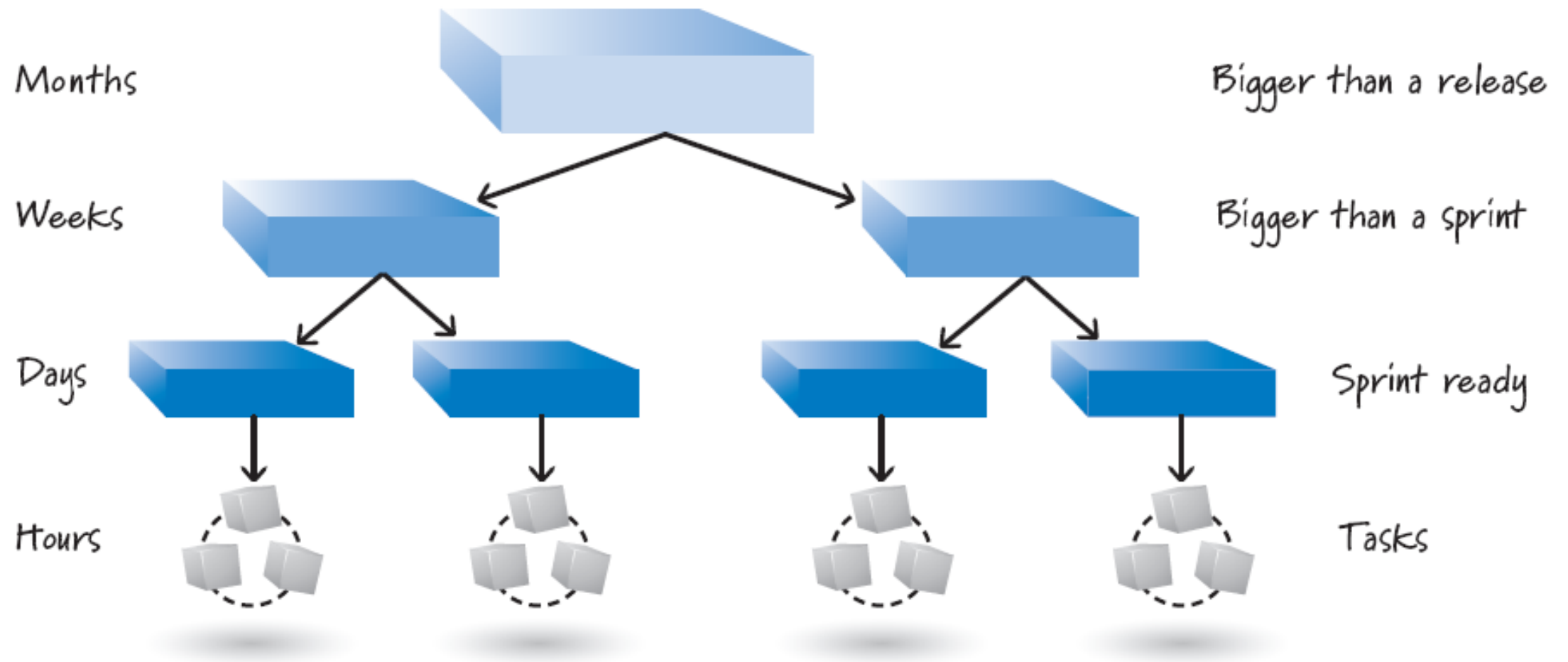
Initially, let's limit uploaded file sizes to be 1 GB or less. Also, make sure that we can properly load common text and graphics files. And for legal reasons we can't have any files with digital rights management (DRM) restrictions loaded to the wiki.

Conversation on the User Story

Conditions of Satisfaction
Verify with .txt and .doc files
Verify with .jpg, .gif, and .png
files
Verify with .mp4 files <= 1 GB
Verify no DRM-restricted files

Confirmation on the User Story

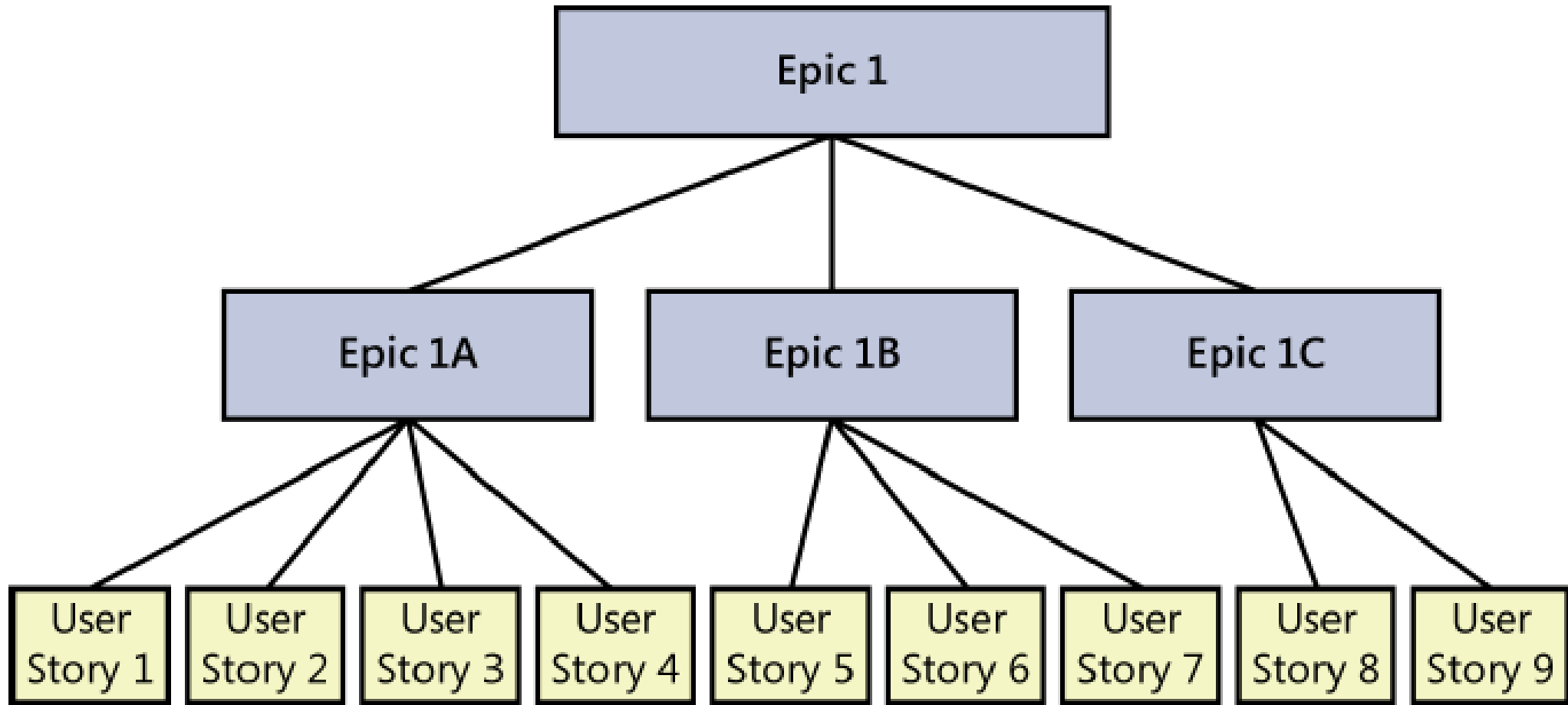
Level of Detail in User Stories



Epic

- are stories that are a few months in size and might span an entire release or multiple releases
- are helpful because they give a very big-picture, high-level overview of what is desired

Epics and User Stories

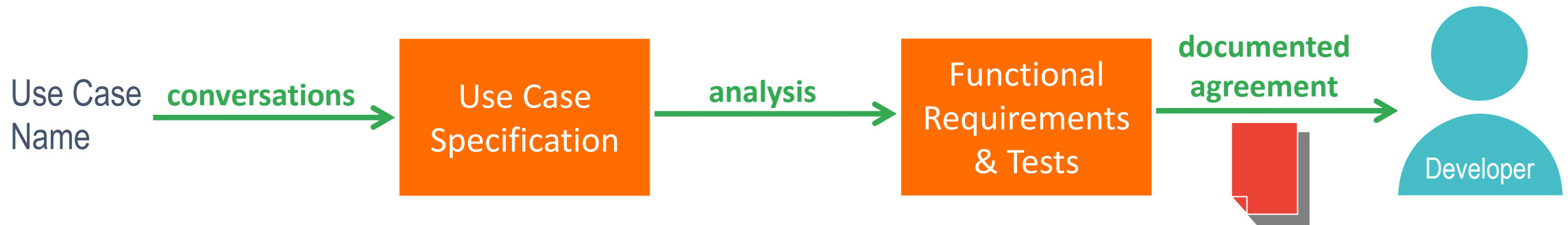


Epic – Example

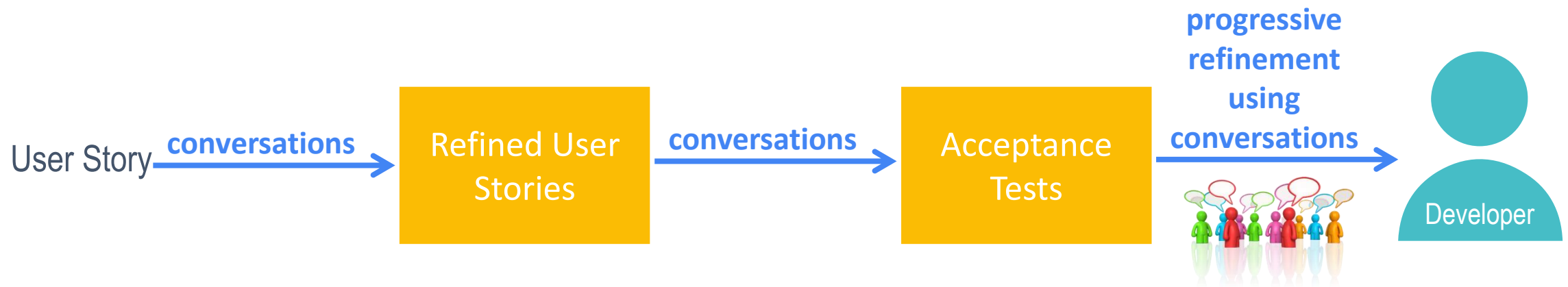
Epic	A user can search for a job
User Stories	<ul style="list-style-type: none">• A user can search for jobs by attributes like location, salary range, job title, company name, and the date the job was posted.• A user can view information about each job that is matched by a search.• A user can view detailed information about a company that has posted a job.

Use Cases & User Stories

“Create an Invoice”



“As a small business owner, I want to create an invoice so that I can bill a customer.”



Unified Modeling Language (UML)



UML

History

a unification and standardization of earlier modeling notations of mostly Booch, Rumbaugh, and Jacobson.

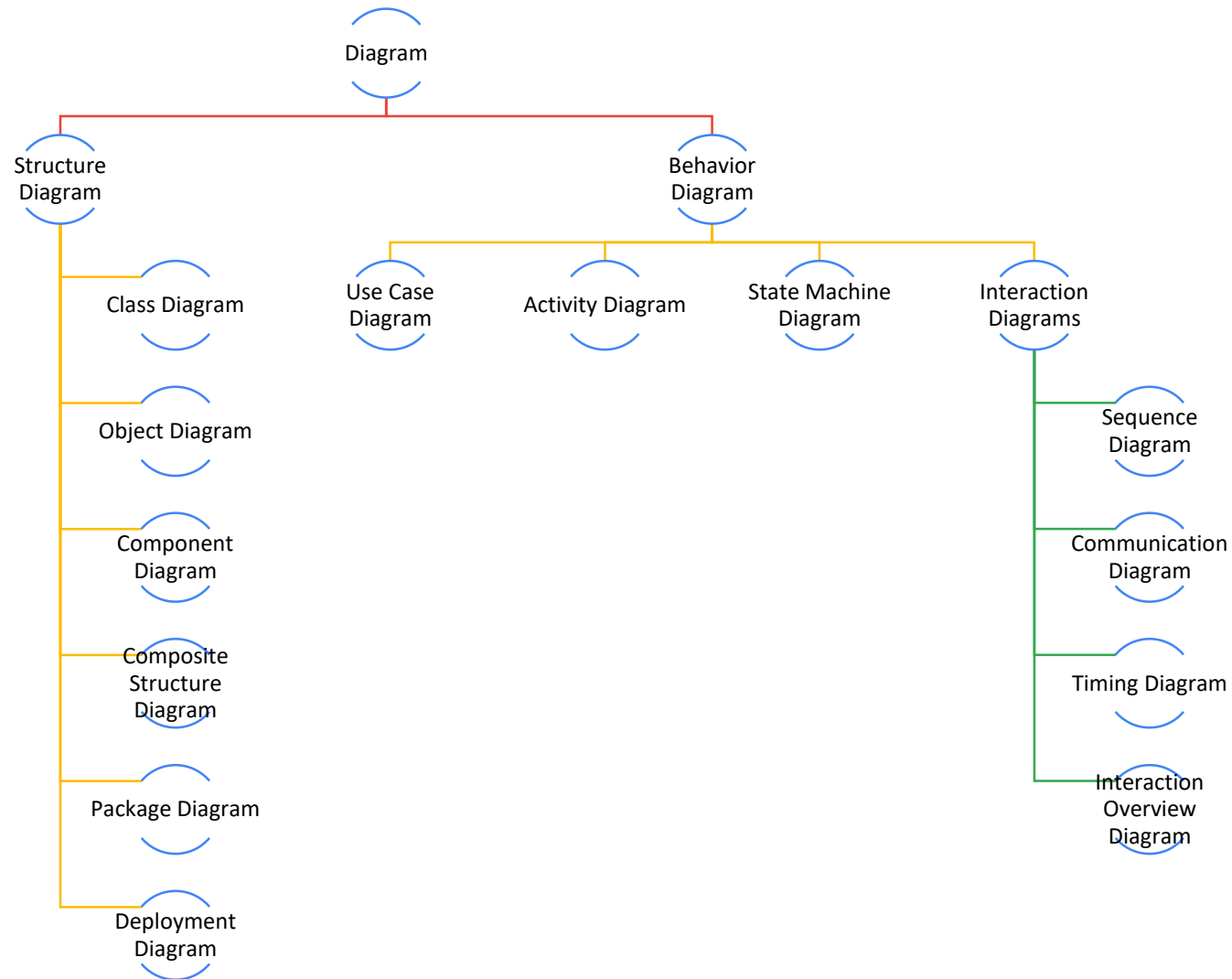
Goal

support efficient communication; prevent misunderstanding and ambiguity.

UML

a visual language for
specifying,
constructing and
documenting
the artifacts of systems

UML Diagram Types



UML in Software Engineering Process

Requirements Analysis

- Use case diagram
- Class diagram
- Activity diagram
- State diagram

Design

- Class diagram
- Sequence diagram
- Package diagram
- State diagram
- Deployment diagram

Documentation

- How much?

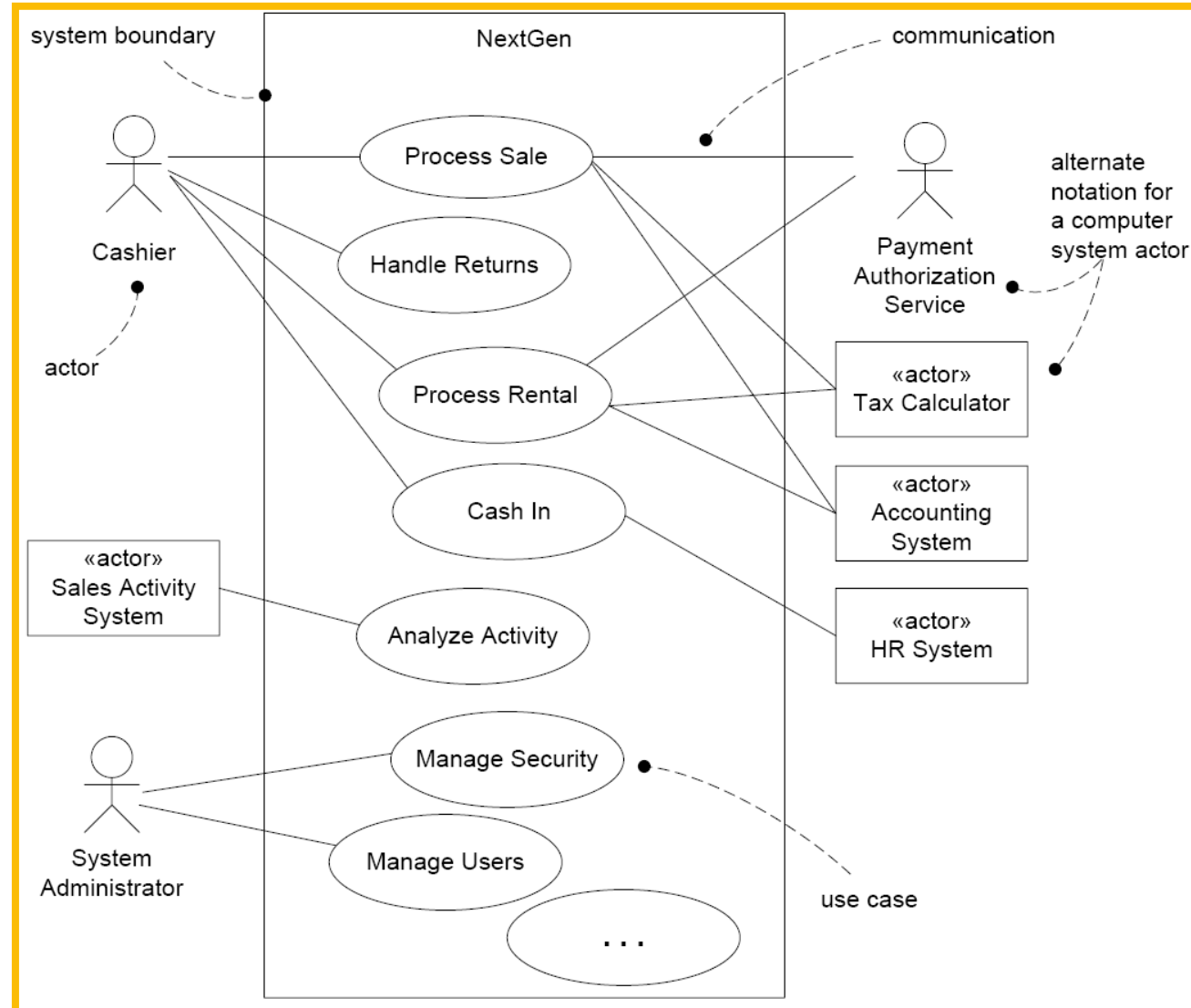
Understanding Legacy Code

- Clarify key points

Use Case Diagram

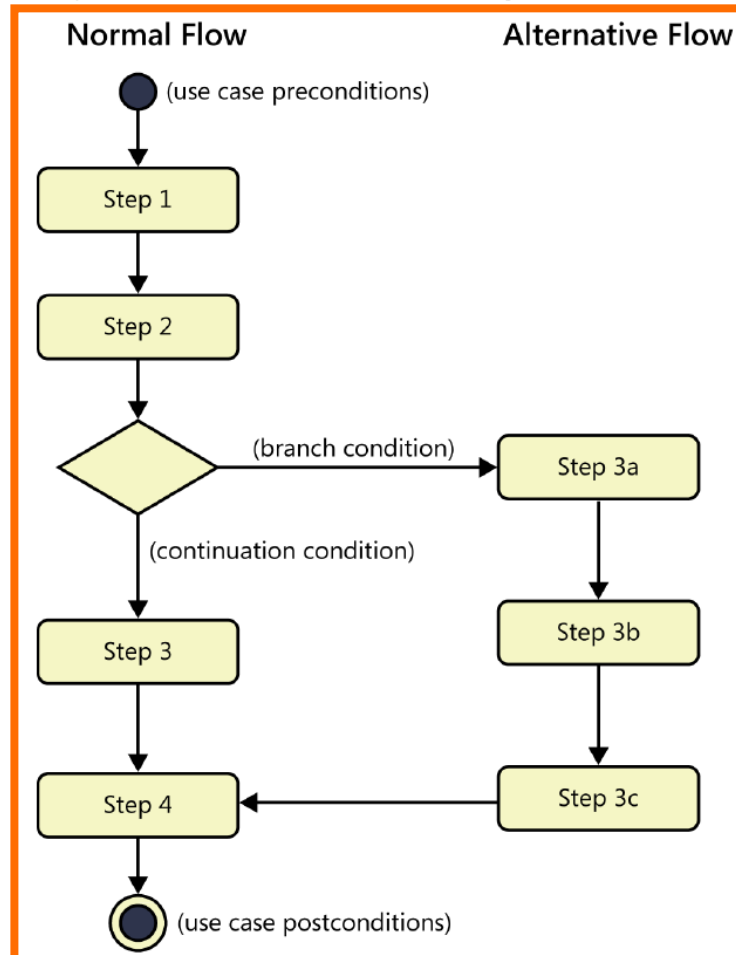
- provides a high-level visual representation of the user requirements
- gives a nice context diagram of a system and its environment
- shows the names of use cases and actors, and their relationships;
provides a quick way to list the use cases by name

Use Case Diagram for POS System

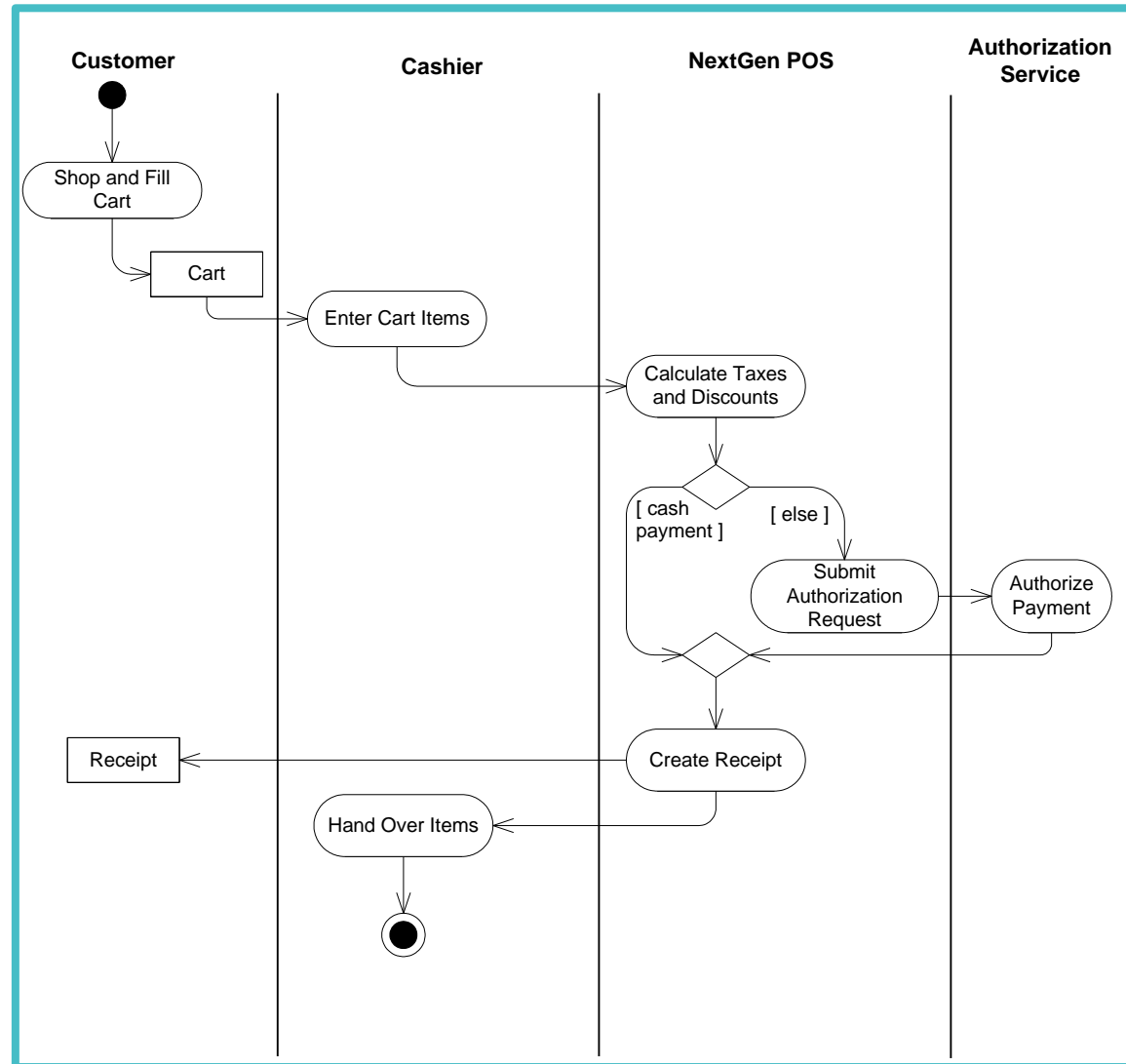


Activity Diagram: Normal Flow and Alternative Flows

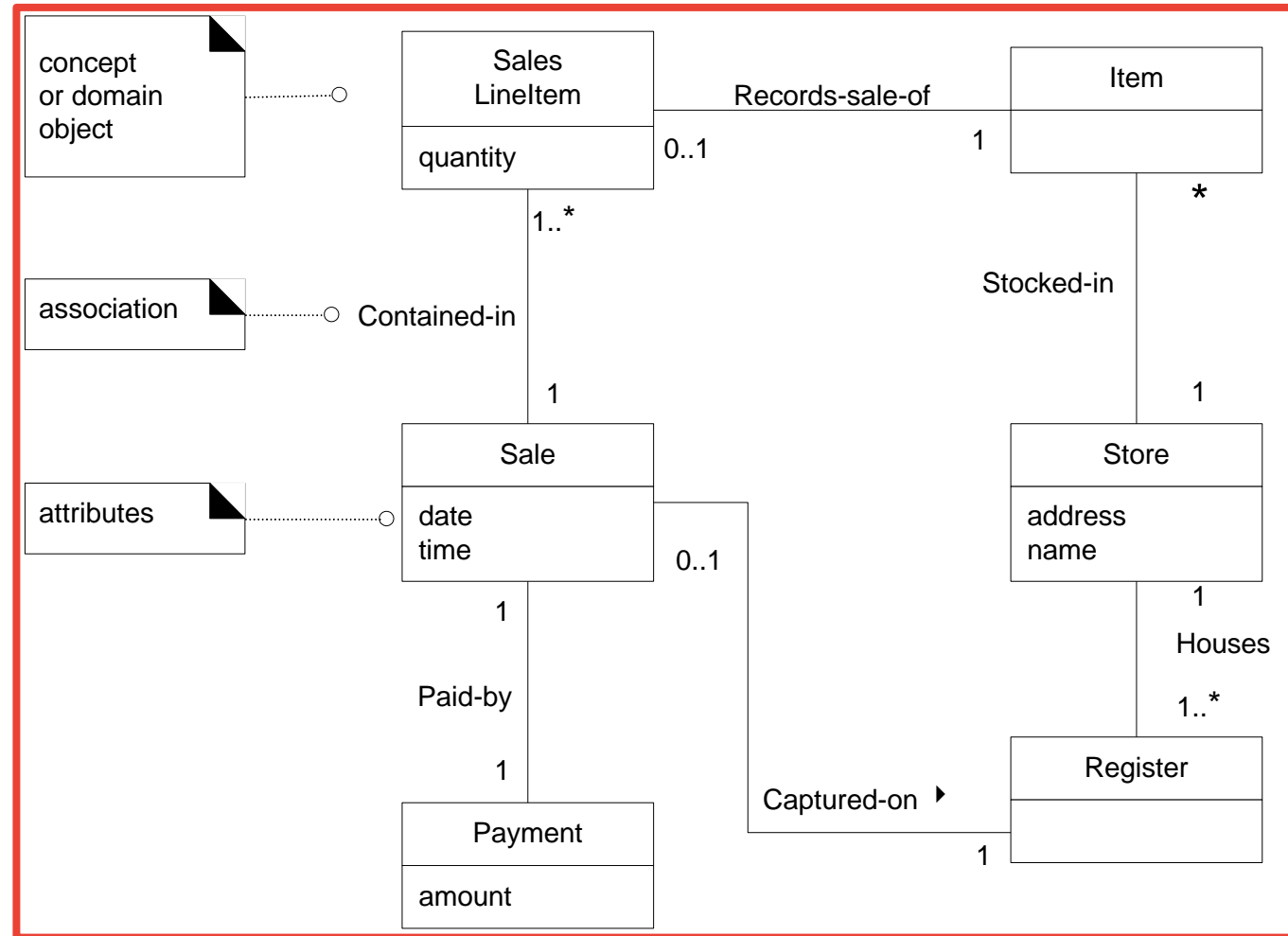
a useful way to visually represent the logic flow in a complex use case



Activity Diagram



Class Diagram



Quality Attributes

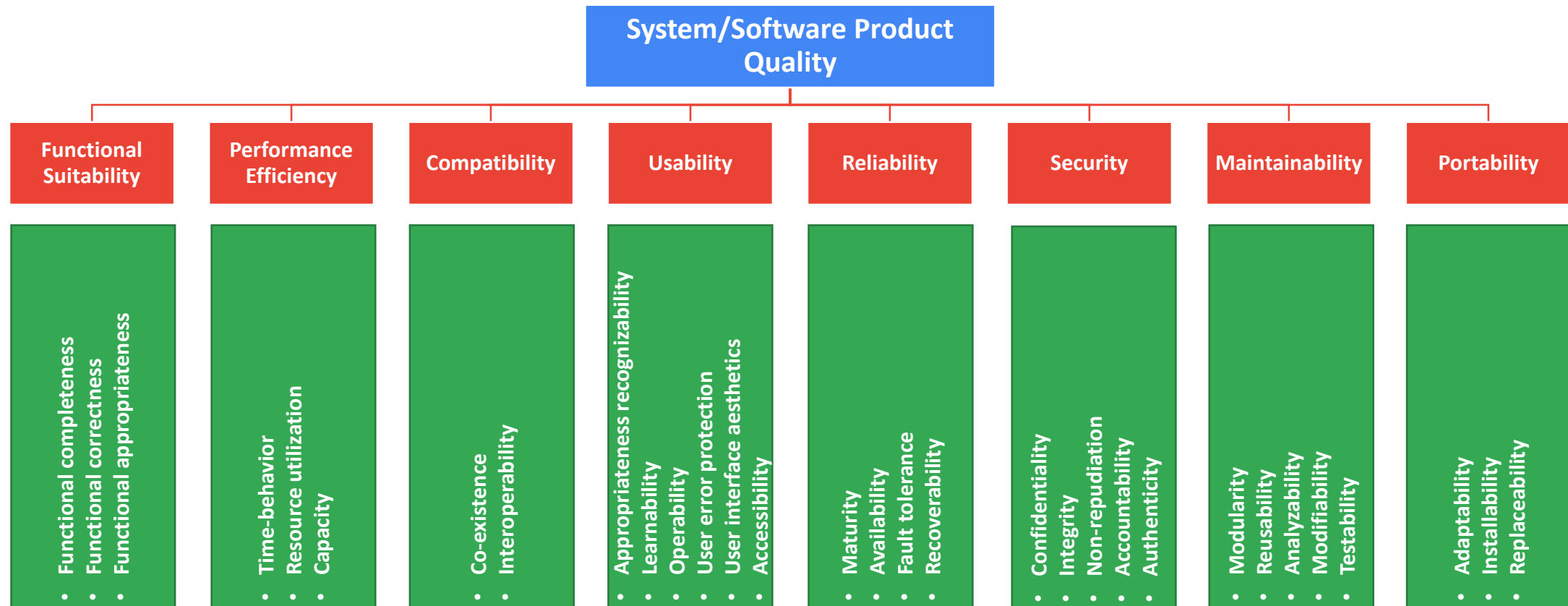
(Non-Functional Requirements)



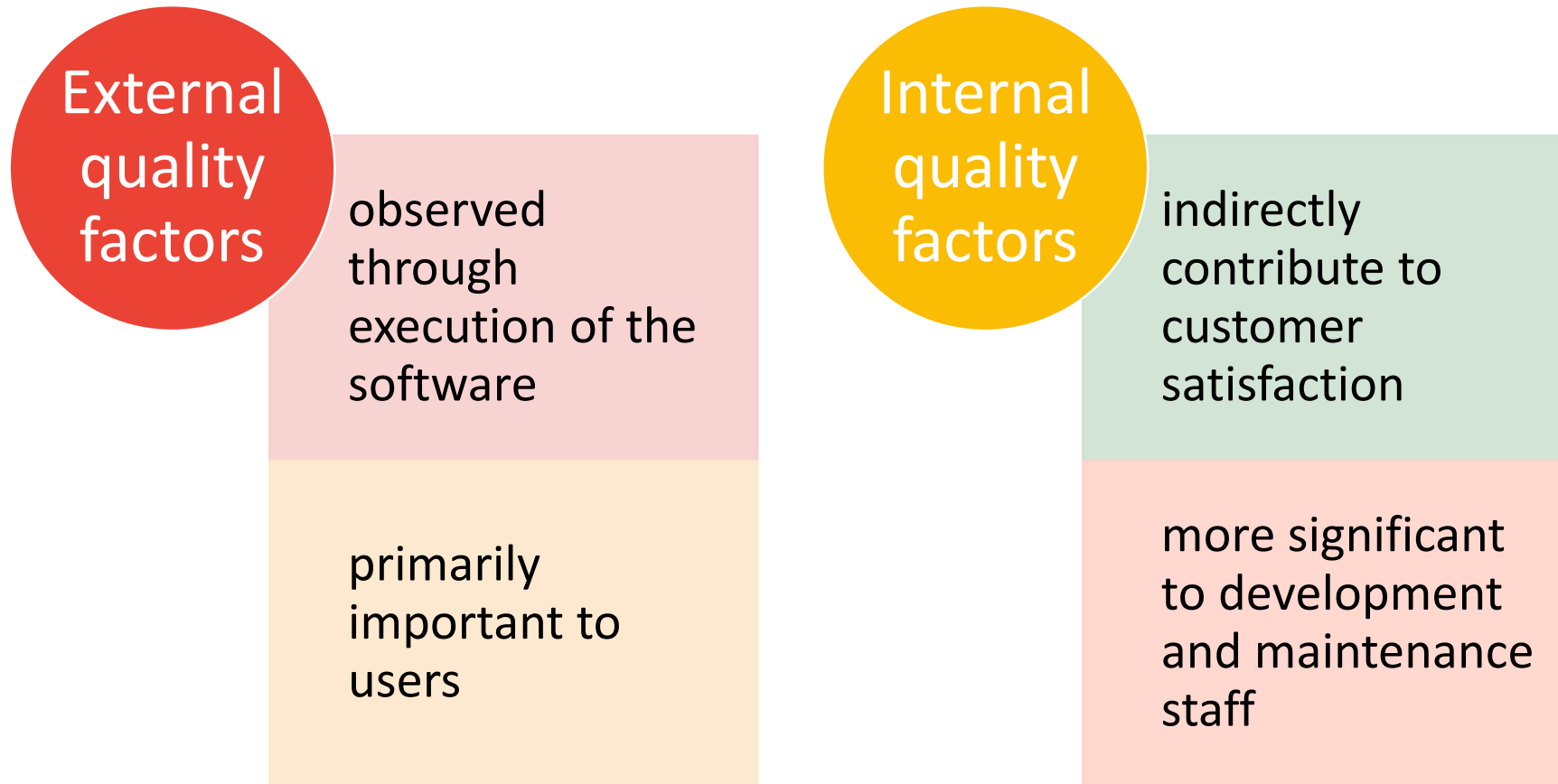
Quality Attributes

- "-ilities" of a system
- qualities of the system, not of the attributes themselves
- not necessarily of high quality
- also important for architectural analysis and design

Product Quality Model – ISO/IEC 25010:2011



Quality Attributes



Some Software Quality Attributes

External quality	Brief description
Availability	The extent to which the system's services are available when and where they are needed
Installability	How easy it is to correctly install, uninstall, and reinstall the application
Integrity	The extent to which the system protects against data inaccuracy and loss
Interoperability	How easily the system can interconnect and exchange data with other systems or components
Performance	How quickly and predictably the system responds to user inputs or other events
Reliability	How long the system runs before experiencing a failure
Robustness	How well the system responds to unexpected operating conditions
Safety	How well the system protects against injury or damage
Security	How well the system protects against unauthorized access to the application and its data
Usability	How easy it is for people to learn, remember, and use the system
Internal quality	Brief description
Efficiency	How efficiently the system uses computer resources
Modifiability	How easy it is to maintain, change, enhance, and restructure the system
Portability	How easily the system can be made to work in other operating environments
Reusability	To what extent components can be used in other systems
Scalability	How easily the system can grow to handle more users, transactions, servers, or other extensions
Verifiability	How readily developers and testers can confirm that the software was implemented correctly

Quality Attributes and Project Types

- **Embedded systems:** performance, efficiency, reliability, robustness, safety, security, usability
- **Internet and corporate applications:** availability, integrity, interoperability, performance, scalability, security, usability
- **Desktop and mobile systems:** performance, security, usability

Availability

- a measure of the planned up time during which the system's services are available for use and fully operational
- is closely related to reliability and is strongly affected by the maintainability subcategory of modifiability
 - *AVL-1. The system shall be at least 95 percent available on weekdays between 6:00 A.M. and midnight Eastern Time, and at least 99 percent available on weekdays between 3:00 P.M. and 5:00 P.M. Eastern Time.*
 - *AVL-2. Down time that is excluded from the calculation of availability consists of maintenance scheduled during the hours from 6:00 P.M. Sunday Pacific Time, through 3:00 A.M. Monday Pacific Time.*

Integrity

- deals with preventing information loss and preserving the correctness of data entered into the system
 - *INT-1. After performing a file backup, the system shall verify the backup copy against the original and report any discrepancies.*
 - *INT-2. The system shall protect against the unauthorized addition, deletion, or modification of data.*
 - *INT-3. The Chemical Tracking System shall confirm that an encoded chemical structure imported from third-party structure-drawing tools represents a valid chemical structure.*
 - *INT-4. The system shall confirm daily that the application executables have not been modified by the addition of unauthorized code.*



Performance

- represents the responsiveness of the system to various user inquiries and actions and more ...

Performance dimension	Example
Response time	Number of seconds to display a webpage
Throughput	Credit card transactions processed per second
Data capacity	Maximum number of records stored in a database
Dynamic capacity	Maximum number of concurrent users of a social media website
Predictability in real-time systems	Hard timing requirements for an airplane's flight-control system
Latency	Time delays in music recording and production software
Behavior in degraded modes or overloaded conditions	A natural disaster leads to a massive number of emergency telephone system calls



Sample Performance Requirements

- *PER-1. Authorization of an ATM withdrawal request shall take no more than 2.0 seconds.*
- *PER-2. The anti-lock braking system speed sensors shall report wheel speeds every 2 milliseconds with a variation not to exceed 0.1 millisecond.*
- *PER-3. Webpages shall fully download in an average of 3 seconds or less over a 30 megabits/second Internet connection.*
- *PER-4. At least 98 percent of the time, the trading system shall update the transaction status display within 1 second after the completion of each trade.*



Security

- deals with blocking unauthorized access to system functions or data, ensuring that the software is protected from malware attacks, and so on
 - User authorization or privilege levels (ordinary user, guest user, administrator) and user access controls
 - User identification and authentication (password construction rules, password change frequency, security questions, forgotten logon name or password procedures, biometric identification, account locking after unsuccessful access attempts, unrecognized computer)
 - Data privacy (who can create, see, change, copy, print, and delete what information)
 - Deliberate data destruction, corruption, or theft
 - Protection against viruses, worms, Trojan horses, spyware, rootkits, and other malware
 - Firewall and other network security issues
 - Encryption of secure data
 - Building audit trails of operations performed and access attempts



Sample Security Requirements

- *SEC-1. The system shall lock a user's account after four consecutive unsuccessful logon attempts within a period of five minutes.*
- *SEC-2. The system shall log all attempts to access secure data by users having insufficient privilege levels.*
- *SEC-3. A user shall have to change the temporary password assigned by the security officer to a previously unused password immediately following the first successful logon with the temporary password.*
- *SEC-4. A door unlock that results from a successful security badge read shall keep the door unlocked for 8.0 seconds, with a tolerance of 0.5 second.*
- *SEC-5. The resident antimalware software shall quarantine any incoming Internet traffic that exhibits characteristics of known or suspected virus signatures.*
- *SEC-6. The magnetometer shall detect at least 99.9 percent of prohibited objects, with a false positive rate not to exceed 1 percent.*
- *SEC-7. Only users who have Auditor access privileges shall be able to view customer transaction histories.*



Usability

- addresses the many factors that constitute what people describe colloquially as *user-friendliness*, *ease of use*, and *human engineering*
 - ease of use
 - ease of learning
 - memorability
 - error avoidance, handling, and recovery
 - efficiency of interactions
 - accessibility
 - ergonomics



Sample Usability Requirements

- *USE-1. A trained user shall be able to submit a request for a chemical from a vendor catalog in an average of three minutes, and in a maximum of five minutes, 95 percent of the time.*
- *USE-2. All functions on the File menu shall have shortcut keys defined that use the Control key pressed simultaneously with one other key. Menu commands that also appear in Microsoft Word shall use the same default shortcut keys that Word uses.*
- *USE-3. 95 percent of chemists who have never used the Chemical Tracking System before shall be able to place a request for a chemical correctly with no more than 15 minutes of orientation.*



Portability

- The effort needed to migrate software from one operating environment to another.
 - *POR-1. Modifying the iOS version of the application to run on Android devices shall require changing no more than 10 percent of the source code.*
 - *POR-2. The user shall be able to port browser bookmarks to and from Firefox, Internet Explorer, Opera, Chrome, and Safari.*
 - *POR-3. The platform migration tool shall transfer customized user profiles to the new installation with no user action needed.*



Scalability

- address the ability of the application to grow to accommodate more users, data, servers, geographic locations, transactions, network traffic, searches, and other services without compromising performance or correctness
 - *SCA-1. The capacity of the emergency telephone system must be able to be increased from 500 calls per day to 2,500 calls per day within 12 hours.*
 - *SCA-2. The website shall be able to handle a page-view growth rate of 30 percent per quarter for at least two years without user-perceptible performance degradation.*
 - *SCA-3. The distribution system shall be able to accommodate up to 20 new warehouse centers.*

Specifying Non-functional Requirements Using User Stories

- affect the design and testing of most or all stories in the product backlog
- can be in "Definition-of-Done" checklist

Internationalization

As a user I want an interface in English, a Romance language, and a complex language so that there is high statistical likelihood that it will work in all 70 required languages.

Web Browser Support

System must support IE8, IE9, Firefox 6, Firefox 7, Safari 5, and Chrome 15.



References

[Cohn]	User Stories Applied: For Agile Software Development, 1st Ed., Mike Cohn, Addison-Wesley Professional, 2004.
[Fowler]	UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd Ed.), Martin Fowler, Addison-Wesley Professional, 2003.
[ISO/IEC 25010:2011]	ISO/IEC 25010:2011 Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models, 2011.
[Jeffries]	Essential XP: Card, Conversation, and Confirmation, Ron Jeffries, XP Magazine, 2001.
[Larman]	Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Ed.), Craig Larman, Prentice Hall, 2004.
[Orso]	"Software Development Life Cycles" course at udacity.com, Alex Orso, 2014.
[Rubin]	Essential Scrum: A Practical Guide to the Most Popular Agile Process, 1st Ed., Kenneth S. Rubin, Addison-Wesley Professional, 2012.
[Wiegers]	Software Requirements, 3rd Ed, Karl Wiegers, Joy Beatty, Microsoft Press, 2013.