What is a process model, what is the purpose of it?
The business owner or the client provides the business requirement in building a software application.

Does the client knows how to build/manufacturer the software?
The owner never knows what has to be done in developing the project, how to develop an software application. So the owner handover the job of building the Software Application to an Software engineering team.

The Engineering team takes requirements, and should turn in an Software System based that meets the business purpose.

How does the engineer team has to build the Software application, what things they have to follow or perform in turning out the requirements into an executable Software System?
They should follow certain process, best-practices and guidelines in building an software application out of the requirements.
how do we know what process, guidelines to be followed?
There are lot of expertise people who has enourmous amount of experience in building and manufacturing software applications in the market, who has identified the guilines and process that can be followed in successfully developing and delivering the applications. So, these people has documented and standardized and provided to the world as Process Models.

Process Model
A software process or methodology is a set of related activities that guides or leads to the production/development of a software system.

There are lot of Sofware process models available and proven in the market which we can directly use in developing a Software application.
1. Waterfall (100 project= 30% projects use to survive, rest of the projects are left without completing)
2. Prototype
3. Incremental & Iterative
4. Spiral
5. Agile (SCRUM)

Let us understand what is waterfall model before moving to SCRUM.
Waterfall is the oldest process model and used to be followed initialy while developing the projects.
The project development process has been divided into 5 phases linearly or sequentially from top the bottom.
1. Requirement Gathering
2. Design
3. Implementation
4. Verification/Testing
5. Release and Maintainance

The development team starts right from the Gathering of requirement, moves down through the phases from top to the bottom util release and maintainance, and there is no way the team can move back during the phases, since it is going from top-to-the bottom sequentially it is called "Waterfall model".

advantages:-
1. quick to understand and easy follow
2. well disciplined we can easily track who is working on what as the whole teams is doing the same
3. fixed phases, so that we can identify at which stage of development we are at.

when to use?
1. project requirements are stable
2. we can produce the complete design of the system initially itself
3. there is guarantee the team process the right design out of the requirement.

drawbacks:-
1. doesnt handle requirement changes well
2. business engagement is high intially and lost over the phases of development
3. if the requirement are interpreted differently then it is not suitable
4. Insufficient time for testing.
5. dont know until we build the system at the end, does the team produce the right product or not
6. whole project has to be planned initially itself

Looks like along with Waterfall model, even other process models suffers from their own dis-advantages either in terms of understanding, complexity, cost or time in delivering. So how to overcome the above process of these process models.

Agile
There are 17 people who belongs to different backgrounds of work/domain/expertise, who has experience on developing systems using different methodologies came together. They came together for brainstorming or discussing to identify a better methodology or model that can be used for building software application quickly with little documentation.

The want to introduce or innovate a process model that well suites for all types of application development, all of a sudden they realized how does one process model serves for any type of application development oops! wrong

17 people has comeup with principles or rules that they want to consider while designing or building the process models, which is called "Agile Manifesto"
"There are 12 principles in agile manifesto"
1. Customer Satisfaction is of highest priority which is achieved through continous delivery of valuable software
2. accomodate change requirements at later phases of development
3. deliver working software frequently
4. business team and development team has to coordinate and colloborate on daily basic
5. higher automony is given to the team members with at most trust
6. Face-to-Face interaction is critical for conveying information to development team.
7. The progress of the project is measured through working software
8. promote development through constant pace

9. team should be having high technical skills
10. simplicity is essential for progress
11. self organizing
12. team should regular meet and think about how to make them more effective

Based on the 12 principle above the team has come up 8 process models, designed based on Agile Methologies
1. Scrum
2. Kabana
3. Extreme Programming
4. Feature Driven Development
5. Adaptive System design
6. Dynamic System development methodology
7. Lean Software development
8. Crystal clear

SCRUM

In Scrum the team is divided into 3 roles
1. Product Owner
2. Scrum Master
3. Scrum Team

#1 Product Owner
1. Business Representative of the Team
2. There are not part-time, should fully engaged for team
3. They should show-up everyday, because there are contributing for the final product.

#2. Scrum Master
1. Most visible spokesperson for the team
2. The scrum master ensures the whole team understands and follows the scrum process in development and delivery of the system.
3. The Scrum Masters are mostly non-technical and helps in the team in implementing the scrum process
4. work to remove any hurdles or blockes out of the way in developing the project.
5. Scrum master draws dashboards, uses charts and graphs in measuring the project status and help them team in tracking and achieving their goals

#3. Scrum Team
1. Team size: 7 plus or minus 2
2. Self organizing
3. Members should be full-time
4. ideally no titles
5. T shaped Resources

#2 Sprint
Follows fail-fast approach, let there be a failure, but let us detect it early so that the damage will be less and rework will be minimal, so that we can get into right track quickly.

What is a process model, why do we need to use it?
Process model is set of steps and guidelines to be followed in developing and delivering the project successfully to its completion. There are many standard process models exists in the market

If we need to develop the application by following any one of these process models, which is called "SDLC" software development lifecycle process of our application
1. Waterfall = linear application development model, where the project development is divided into 5 phases and followed sequentially from top-to-bottom in developing the application.
Phase: (Requirement Gathering, Design, Implementation, Verification and Maintainance)
2. Prototype = build prototype demonstrate the working model, based on which build actual system
3. Incremental & Iterative model = build the system in smaller groups or units (in phases)
4. Spiral model = For each requirement apply all the phases individually and build the system.

5. Agile methodologies
Agile Methodology has designed Agile Manifesto which comprises of 12 principles based on which various different process models are defined addressing different types of application development
"12" principles of agile manifesto
1. high customer satisfaction by quickly releasing the product
2. accomodate changes
3. business and development team has to meet on daily basis
4. face-to-face interaction is required for coveying information within a development team
5. progress is measured through working software
6. highest autonomy is given to individual remember with more trust
7. Simplicity is essential
8. Self-Organizing teams are required
9. Constant development in fixed pace
10. The team should reflect how to become more effective
11. technical excellence and good design should be the main focus
12. deliver software frequently

Based on the above principle the Agile Software process models are built:
1. Scrum
2. Kabana
3. Extream programming
4. Crystal Clear
5. Dynamic System development
6. Feature driven development
7. Lean programming
8. Adaptive System design

Scrum is one of the such popular process models of the Agile methodology.
The are 3 groups of people work in Scrum

1. Product Owner
  1.1 Business Representative from the client
  1.2 Should be fulltime
  1.3 should appear daily, because he has to contribute for the final product

2. Scrum Master
  2.1 most visible person within team
  2.2 mentors and guides the team in enforcing the scrum practices
  2.3 devices charts, graphs and dashboards helping the team understand the progress the System
  2.4 The first point of escalation
  2.5 Mostly non-technical
  2.6 blockers or hurdles should be brought up to the scrum master, so that he resolve them in continuing the development

3. Scrum Team
  3.1 Team size should be of 7, which can be plus/minus 2, more team member proven to be in-efficient in handling
  3.2 all the members should be fulltime (exception to few people: dba, system engineers etc)
  3.3 no titles
  3.4 self organized
  3.5 T-Shaped Resource

Vision

Setting up the Vision for the project
Vision tells the team about the system we want to develop and destination we want to reach
In-short: What we want to achieve, or build is called "Vision"

The destination contained within the vision is what call Minimum Viable product or MVP that can be quickly released out of the door to early adopters, so that based on feedback we can build further.

For eg.. Vision
  Build a platform where property owner should be able to place their property information on the platform market them, so that buyers can search for property based on their requirement and should reach to the owners in buying the property

Themes

Identifying the Themes
The first-level of decomposition is to identify the Themes, themes are nothing but similar group of work items, that can analyzed and can be developed to gether easily. we can treat them similar to modules
  From the above Vision we can derive the below Themes.
    1. User Management
    2. My Account
    3. Properties
    4. Search
    5. Payment
    6. Support

Themes can be further decomposed into Features, which are the next smaller piece of work that has to developed
For eg.. from User Management Themese we can derive below features:
  1.1 Login
  1.2 Registration
  1.3 Forgot Password
  1.4 Change Password
  1.5 Remember Me

The features we can derive from My Account
  2.1 Edit Profile
  2.2 My Properties
  2.3 Favourites
  2.4 Recent Conversations
  2.5 Addresses

The product owner may say only features we need for this release are login and registration only, others are nice to have.

User Story:-
The people who are going to use the System are customers and end-users, we need to describe the system requirements from the point of how the user is going to interact with product and use it, as we describe the System from the user perspetive it is called User Story.
unless the level details describing from the point of user is presented we cannot build and delivery the system.
Most of the time product owner is going to write the user story, apart from him any other member within the team also can contribute, while writing the user story we should keep mind the user story adds a value from the end-user point of view to the system.

While writing the user story we need to follow Bill Wakes Invest acronym guide.
  INDEPENENT = Should be indepenent of other user stories
  NEGOTIABLE = Flexible scope, explain the intention not the implementation
  VALUABLE = should add value to the system
  ESTIMABLE = describe to the detail, so that people can identify/quatify the user story
  SMALL = each story should be described in such a way, where it can developed, tested and delivered in one interaction
  TESTABLE = capture details information, so that people can identify how to test also

Task

Once everyone understands clearly their user stories of the Sprint, the team has to write tasks needed to for each story to complete.

Theme: User Management
  Feature: Registration
    User Stories:-
      1. Present the user a form where he/she has to enter the details of registration and submit
      2. System should send an email verification link asking the user

to verify the email address
3. An otp has to be generated during the registration process
4. User should be presented with an page, asking to enter the otp sent to verify the mobile number
5. upon completing the email verification link system should display a confirm page making understand the registration completion asking him to proceed for login
6. An welcome email onboarding the user should be send upon completing the all verification steps

The developer is responsbile for adding the tasks describing the work that should be done for completing the user story.
For eg. tasks for the below user story
1. Present the user a form where he/she has to enter the details of registration and submit
1.1 identify the database tables into which we need to persist the data and capture master data into each of these tables for facilitating the registration process
1.2 build the user interface pages asking the user to fill the form and submit the details
1.3 build an user interface page to display the confirmation page showing up to the user
1.4 write the code for inserting the data into database tables in dao component
1.5 write the service tier to perform buisness logic and invoke dao
1.6 write the controller component in handling enduser interaction with the system

Each task has to be clearly estimated
Now we know the total amount of time required for completing the user stories, we need to identify the capacity of the team to deliver

Estimation:-
There are 2 types of estimates
1. Actual Estimates = will gives you the exact amount of time/hours that takes in completing a task
2. Relative Estimates = In comparing with our past experiences in working up a similar type of work item we give relative estimates quantifying in terms of size.

For each user story we picked up in Sprint planning meet the team has to give estimate. Since we dont know what tasks has to be performed in accomplishing the user story we cannot expect actual estimates here rather we do relative estimation.

The relative has to be given based on the past experience in dealing with a similar type of user story and quatify it to be small, medium or big.

There are lot of estimation technics are available
1. planning poker
2. story points
3. Focus factory
4. dirty hours
5. mandays

Each and every user story should be quatified in terms of story points which is expressed in fibinocci series 1 2 3 5 8 13 21 >

1, 2, 3 = small story
5, 8 = medium complex
13, 21 = highly complex
above 21 = very difficult to develop and test as well

Now once the user story has been assigned to the sprint based on story points(relative estimate), now the scrum member has to write tasks in completing the story and has to give actual estimates to each task. the sum of all the task estimates becomes the actual estimate of the story to complete

Epic:

Epic are the users stories which are not yet defined and kept for future sprints. The entire system we want to build will be divided into EPIC, these are just like planned items which we want build and but not described yet detailed

We can say a rough roadmap or a vogue plan in how we want the system to be build.

RoadMap = prioritizing the Themes to be worked up within the EPIC
Release Plan = Number of Sprints to be delivered together is called a Release plan

---
