
Software Engineering
Course Code : 210253

Course Objectives

- To learn and understand the principles of software engineering
- To be acquainted with methods of capturing, visualising and analysing software requirements
- To apply design and testing principles to software project management
- To understand project management through life of the project

Course Outcomes

- C01: Analyze software requirements and formulate design solution for a software.
- CO2: Design applicable solutions in one or more application domains using software engineering approaches that integrate ethical, social, legal and economic concerns.
- CO3: Apply new software models, techniques and technologies to bring out innovative and novelistic solutions for the growth of the society in all aspects and evolving into their continuous professional development.
- C04: Model and design User interface and component-level.
- CO5: Identify and handle risk management and software configuration management.
- C06: Utilize knowledge of software testing approaches, approaches to verification and validation.
- C07: Construct software of high quality — software that is reliable, and that is reasonably easy to understand, modify and maintain efficient, reliable, robust and cost-effective software solutions.

Books

- 1. Roger Pressman, —Software Engineering: A Practitioner‘s Approachǁ, McGraw Hill, ISBN 0–07–337597–7
- 2. Ian Sommerville, — Software Engineeringǁ, Addison and Wesley, ISBN 0-13-703515-2

- Module 1: Introduction to Software Engineering, Software Process Models

Sinhgad Institutes

Dr. Manjusha S. Joshi

Objectives

- Software Engineering Fundamentals:
  - Introduction to software engineering,
  - The Nature of Software, Defining Software, Software Engineering Practice.
- Software Process:
  - A Generic Process Model,
  - defining a Framework Activity,
  - Identifying a Task Set,
  - Process Patterns,
  - Process Assessment and Improvement,
  - Prescriptive Process Models,
  - The Waterfall Model,
  - Incremental Process Models,
  - Evolutionary Process Models,
  - Concurrent Models,
  - A Final Word on Evolutionary Processes.
  - Unified Process,
  - Agile software development: Agile methods, plan driven and agile development.

Introduction – What is Software?

- Software is:
  - Instructions (computer programs) that when executed provide desired features, function, and performance
  - Data structures that enable the programs to adequately manipulate information
  - Documentation that describes the operation and use of the programs
  - Software is both a product and a vehicle that delivers a product(e.g. Operating System)

What is Software ?

Software is a product

- Transforms information - produces, manages, acquires, modifies, displays, or transmits information
- Delivers computing potential of hardware and networks

Software is a vehicle for delivering a product

- Controls other programs (operating system)
- Effects communications (networking software)
- Helps build other software (software tools & environments)

What is Software ?

Software products may be developed for a particular customer or may be developed for a general market.

- Software products may be
  - **Generic** - developed to be sold to a range of different customers e.g. PC software such as Excel or Word.
  - **Bespoke** (**custom**) - developed for a single customer according to their specification. E.g. air traffic monitoring

Definition of software

- Software is ..
  - **Instructions that when executed provide desired features, function and performance**
  - **Data structures that enable the program to manipulate information.**
  - **Documentations or descriptive information in hard copy and virtual forms that describes the operation and use of programs**

Software Engineering Definition

- The IEEE definition:
  - Software Engineering: The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
  - Engineering approach to develop software.
    - Building Construction Analogy.
  - Systematic collection of past experience:
    - techniques,
    - methodologies,
    - guidelines.

A Layered Technology

- Software engineering is a layered technology

- Any engineering approach (including software engineering) must rest on an organizational commitment to quality.
- TQM, Six Sigma or any similar philosophies foster a continuous process improvement culture, and it is this culture that ultimately leads to the development of increasingly more effective approaches to software engineering.
- The bedrock that supports software engineering is a quality focus

Layered Technology

- Foundation layer is Process Layer
- Definition of Process : a series of actions or steps taken in order to achieve a particular end.
- Process Model : A framework i.e. responsible for milestones, quality, work products
  - Methods: Technical — "how to" s. Methods encompass a broad array of tasks that include communication, requirements analysis, design modeling, program construction, testing, and support.
- Tools : Automated or simulated support for s/w development. When tools are integrated so that information created by one tool can be used by another, a system for the support of software development, called *computer-aided software engineering, is established.*

Software Process

- A process is a collection of activities, actions and tasks to create a work product.
- An *activity strives to achieve a broad objective* (e.g., communication with stakeholders) and is applied regardless of the application domain, size of the project, complexity of the effort
- An *action (e.g., architectural design) encompasses* a set of tasks that produce a major work product (e.g., an architectural design model)
- A *task focuses on a small, but well-defined objective (e.g., conducting a unit* test) that produces a tangible outcome

Software Process

- A process is *not a rigid prescription for how* to build computer software.
- The intent is always to deliver software in a timely manner and with sufficient quality to satisfy those who have sponsored its creation and those who will use it.
- Includes umbrella activities which protect and control the process.
- Process Framework has 5 activities:
  - Communication
  - Planning
  - Modeling
  - Construction
  - Deployment

Process Framework

- **Communication :** The intent is to understand stakeholders' objectives for the project and to gather requirements that help define software features and functions.
  - Heavy communication with customers, stakeholders, team
  - Encompasses requirements gathering and related activities
- **Planning :** defines the software engineering work by describing the technical tasks to be conducted, the risks that are likely, the resources that will be required, the work products to be produced, and a work schedule.
  - Workflow that is to follow
  - Describe technical task, likely risk, resources will require, work products to be produced and a work schedule

Process Framework

- **Modeling :** A software engineer create models to better understand software requirements and the design that will achieve requirements.
  - Help developer and customer to understand requirements (Analysis of requirements) & Design of software
- **Construction :** This activity combines code generation (either manual or automated) and the testing that is required to uncover errors in the code
  - Code generation: either manual or automated or both
  - Testing – to uncover error in the code
- **Deployment** : The software (as a complete entity or as a partially completed increment) is delivered to the customer who evaluates the delivered product and provides feedback based on the evaluation
  - Delivery to the customer for evaluation
  - Customer provide feedback

Software Process

- **Software project tracking and control**—allows the software team to assess progress against the project plan and take any necessary action to maintain the schedule.
- **Risk management**—assesses risks that may affect the outcome of the project or the quality of the product.
- **Software quality assurance**—defines and conducts the activities required to ensure software quality.
- **Technical reviews—**assesses software engineering work products in an effort to uncover and remove errors before they are propagated to the next activity

Software Process

- **Measurement—**defines and collects process, project, and product measures that assist the team in delivering software that meets stakeholders' needs; can be used in conjunction with all other framework and umbrella activities.
- **Software configuration management—** manages the effects of change throughout the software process.
- **Reusability management**—defines criteria for work product reuse (including software components) and establishes mechanisms to achieve reusable components.
- **Work product preparation and production** —encompasses the activities required to create work products such as models, documents, logs, forms, and lists.

Software Engineering Practice

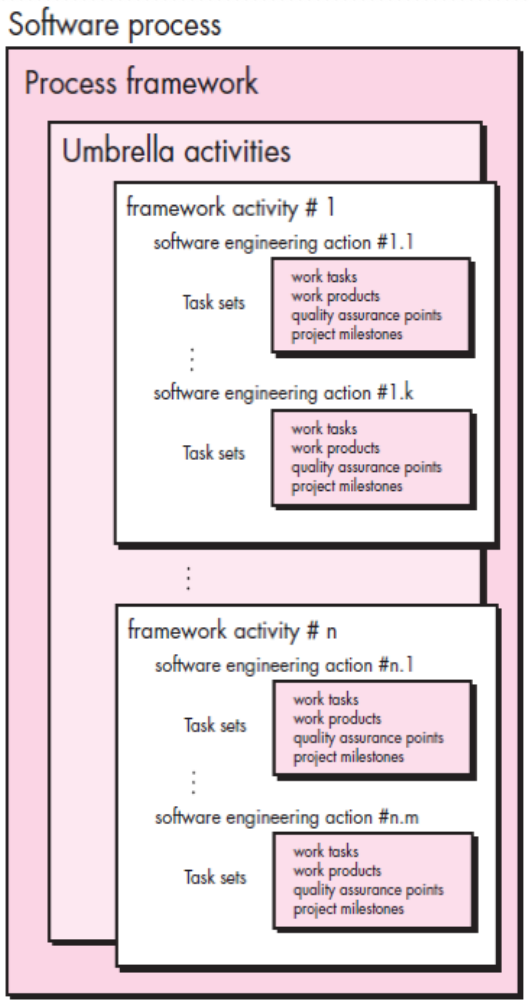
- Understand the problem(communication and analysis)
- Plan the solution (modeling and software design).
- Carry out the plan (code generation).
- Examine the result(testing and quality assurance).

Generic Process Model

- Process is a collection of activities, actions and tasks for a work product to be created.

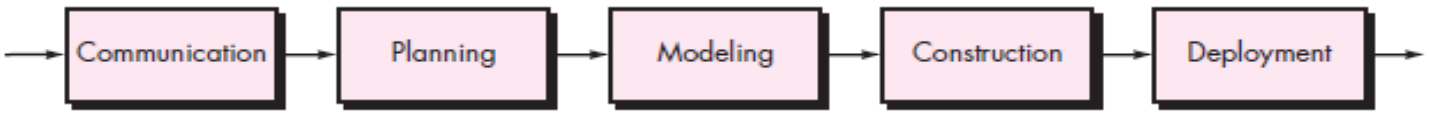
All these reside within a framework or a model that define the relationship with the process and with one another. Software Process is represented in the figure.

There can be linear, iterative, evolutionary or parallel process flows executing framework activities.

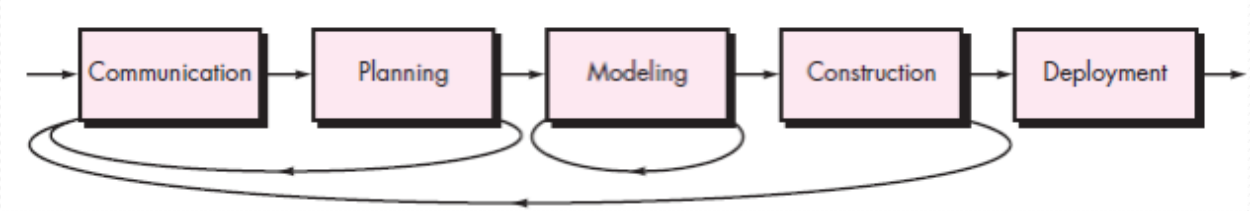


Process Flow

- Linear Process Flow
  - Executes flow from communication to deployment in sequence

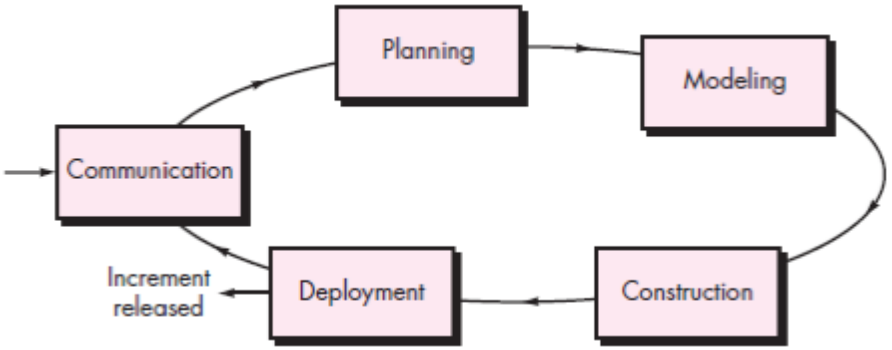


- Iterative Process Flow



Process Flow

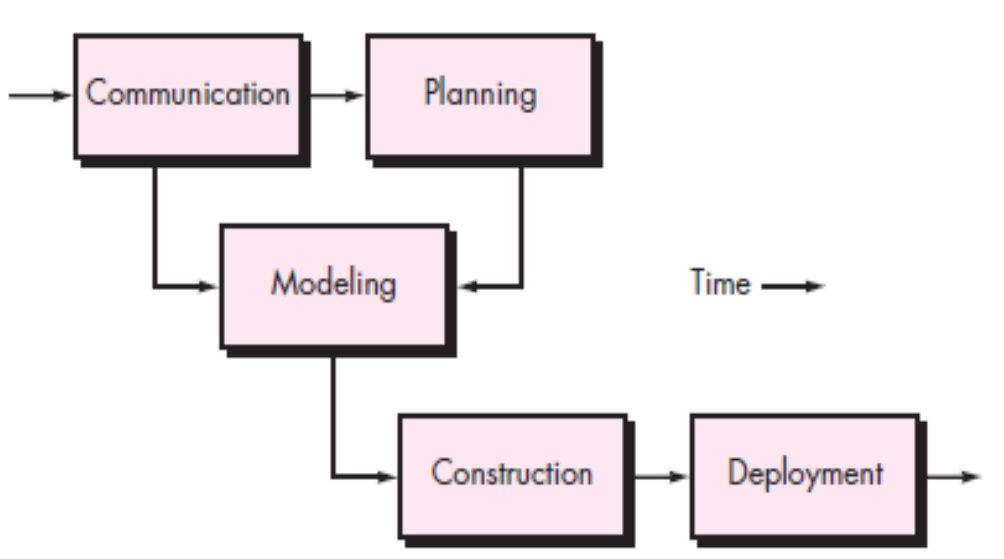
- Evolutionary Process Flow
  - Executes the activities in a —circular manner to provide a software



- Parallel Process Flow
  - Executes one or more activities in parallel with other activities

Process Flow

- Parallel Process Flow



Types of Process Models

- Generic Process Model
- Prescriptive Process Model
- Evolutionary Process Model

Generic Process Model

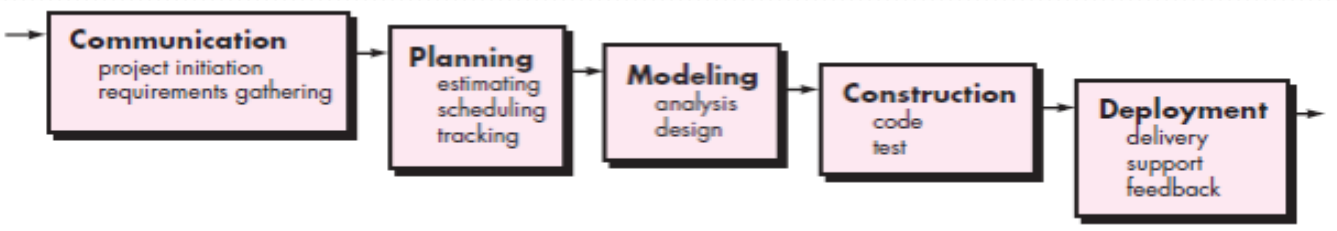
- A generic process model has:
- 1.Defining a framework activity:
  - Work tasks for a simple software project are:
    - Make contact with stakeholder via telephone.
    - Discuss requirements and take notes.
    - Organize notes into a brief written statement of requirements.
    - E-mail to stakeholder for review and approval.
  - If Project is complex, then go for: inception, elicitation, elaboration, negotiation, specification, and validation.
- 2.Identifying a task set:
  - Task sets are actual list of work to be done to accomplish the objective of s/w engg. action.
  - Select suitable task set based on problem and project.
- 3.Process patterns: Proven solutions for the similar problems

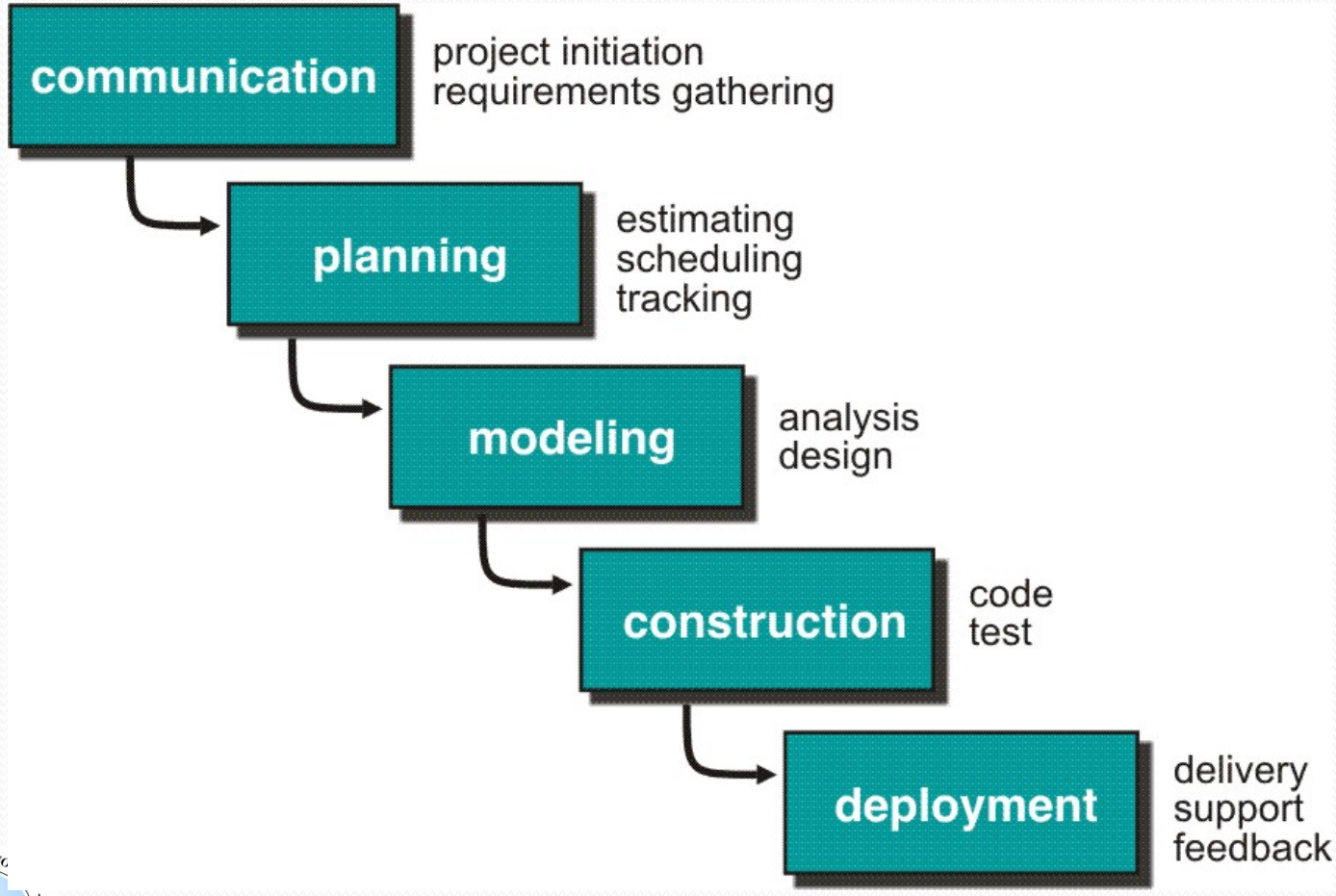
Prescriptive Process Models

- Prescriptive process models define a prescribed set of process elements and a predictable process work flow.
- Organize framework activities in a certain order
- Process framework activity with set of software engineering actions.
- Each action in terms of a task set that identifies the work to be accomplished to meet the goals.
- The resultant process model should be adapted to accommodate the nature of the specific project, people doing the work, and the work environment.

Prescriptive Process Models

- The Waterfall Model : a systematic, sequential approach to software development that begins with customer specification of requirements and progresses through planning, modeling, construction, and deployment, culminating in ongoing support of the completed software
- Calling this model as "Prescribe" because it recommend a set of process elements, activities, action task, work product & quality.
- Each elements are inter related to one another (called workflow).




communication
project initiation
requirements gathering
planning
estimating
scheduling
tracking
modeling
analysis
design
construction
code
test
deployment
delivery
support
feedback

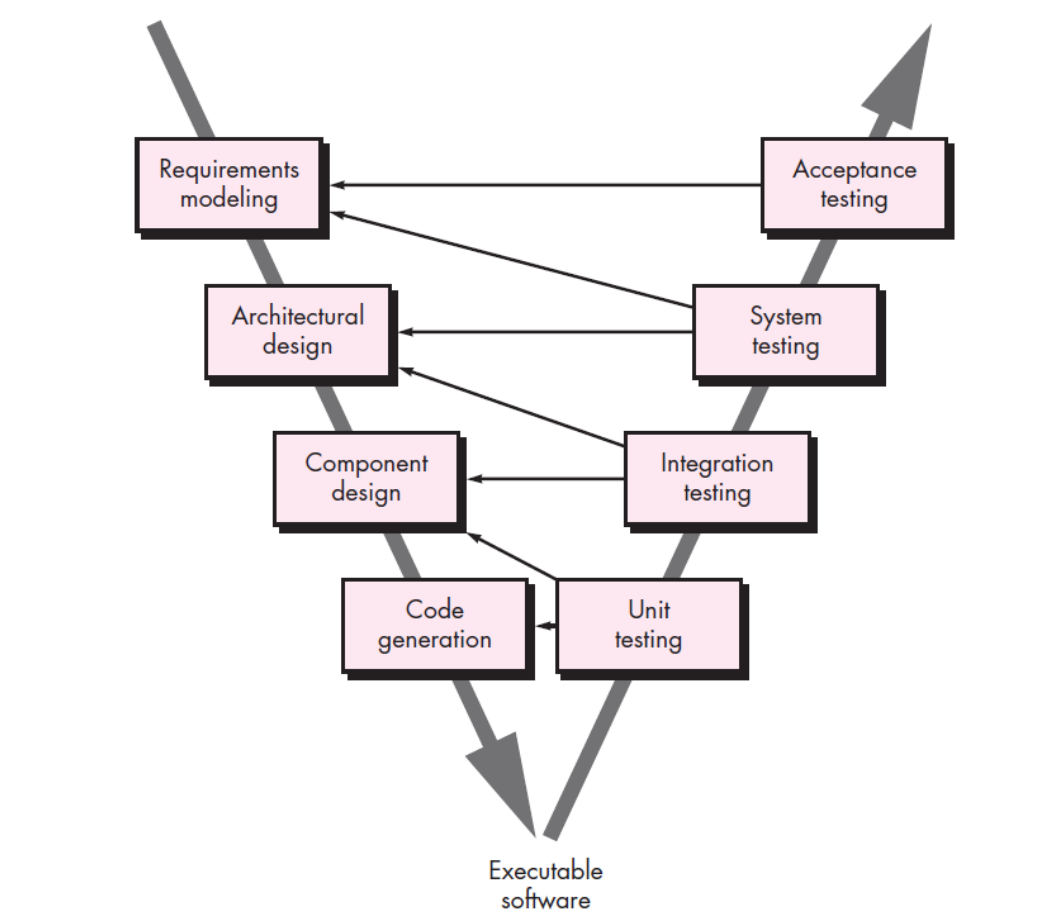
Waterfall Model or Classic Life Cycle

- Requirement Analysis and Definition: What - The systems services, constraints and goals are defined by customers with system users.
- Scheduling tracking -
  - Assessing progress against the project plan.
  - Require action to maintain schedule.
- System and Software Design: How –It establishes and overall system architecture. Software design involves fundamental system abstractions and their relationships.

Waterfall Model or Classic Life Cycle

- Integration and system testing: The individual program unit or programs are integrated and tested as a complete system to ensure that the software requirements have been met. After testing, the software system is delivered to the customer.
- Operation and Maintenance: Normally this is the longest phase of the software life cycle. The system is installed and put into practical use. Maintenance involves correcting errors which were not discovered in earlier stages of the life-cycle.

Waterfall Model



Waterfall Model

- Disadvantages:

- The nature of the requirements will not change very much During development; during evolution
- The model implies that you should attempt to complete a given stage before moving on to the next stage
  - Does not account for the fact that **requirements constantly change.**
  - It also means that customers can not use anything until the entire system is complete.
- The model implies that once the product is finished, everything else is maintenance.
- Surprises at the end are very expensive
- Some teams sit ideal for other teams to finish
- Therefore, this model is only appropriate when the requirements are well-understood and changes will be fairly limited during the design process.

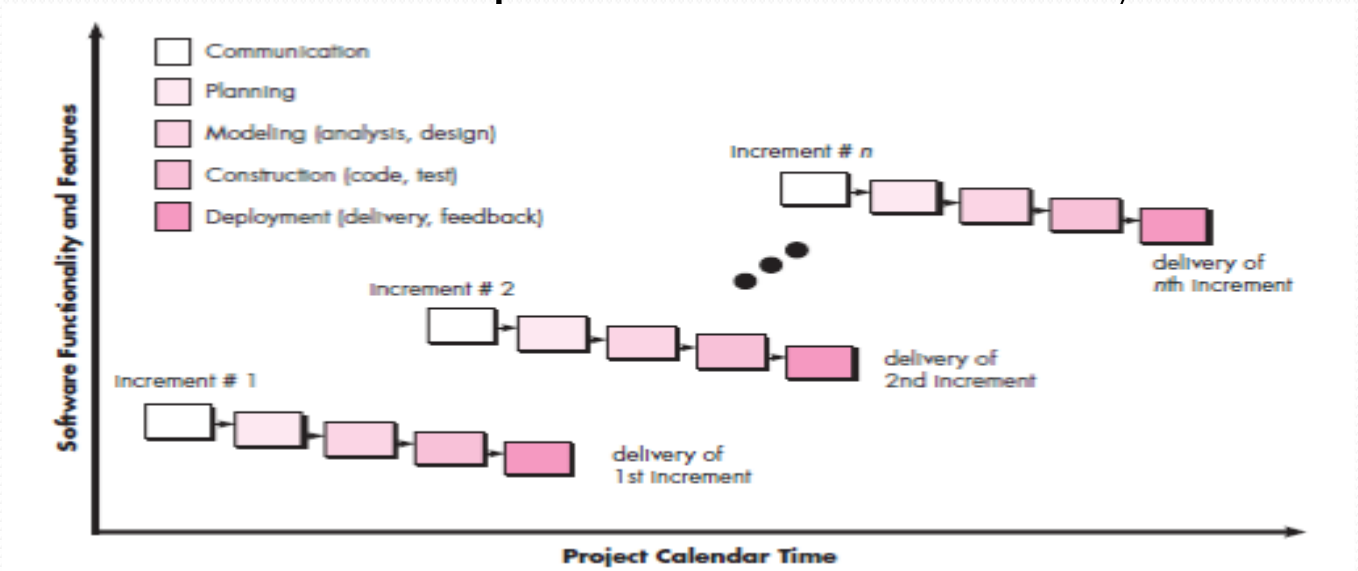
Problems:

- Real projects are rarely follow the sequential model.

1. Difficult for the customer to state all the requirement explicitly.
2. Assumes patience from customer - working version of program will not available until programs not getting change fully.

Incremental Process Model (Ex: The RAD Model)

- This is used when there is a need to provide limited set of software functionality to users quickly and then refine and expand that functionality in later software



Delivers software in small but usable pieces, each piece builds on pieces already delivered

Incremental Process Model (Ex: The RAD Model)

- When an incremental model is used, the first increment is often a *core product*
- The incremental process model focuses on the delivery of an operational product with each increment

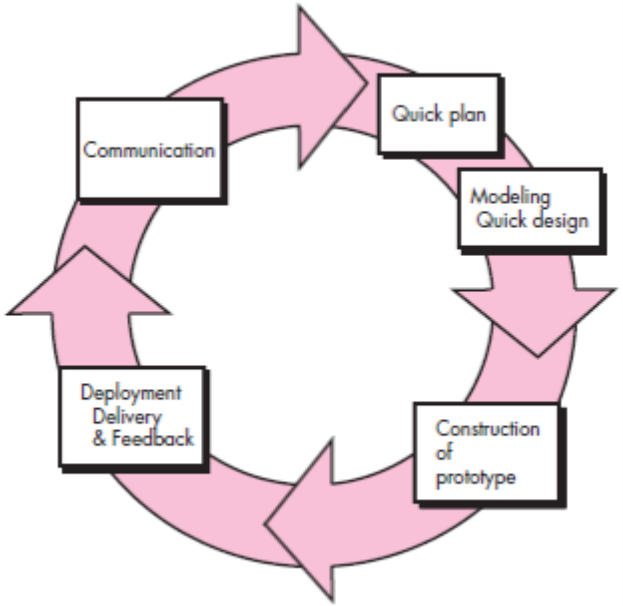
- Staffing, delivery date and resources are managed.
- Disadvantages:
  - Each phase is rigid and no overlapping.
  - Not all requirements are gathered up at once. Problems may arise

Evolutionary Process Models

- Evolutionary process models produce an increasingly more complete version of the software with each iteration.
- They are different from incremental model as the requirements keep on changing here.
- Evolutionary models are iterative.
- Examples include :
  - Prototyping
  - Spiral Model
  - Concurrent Development Model

Evolutionary Process Models

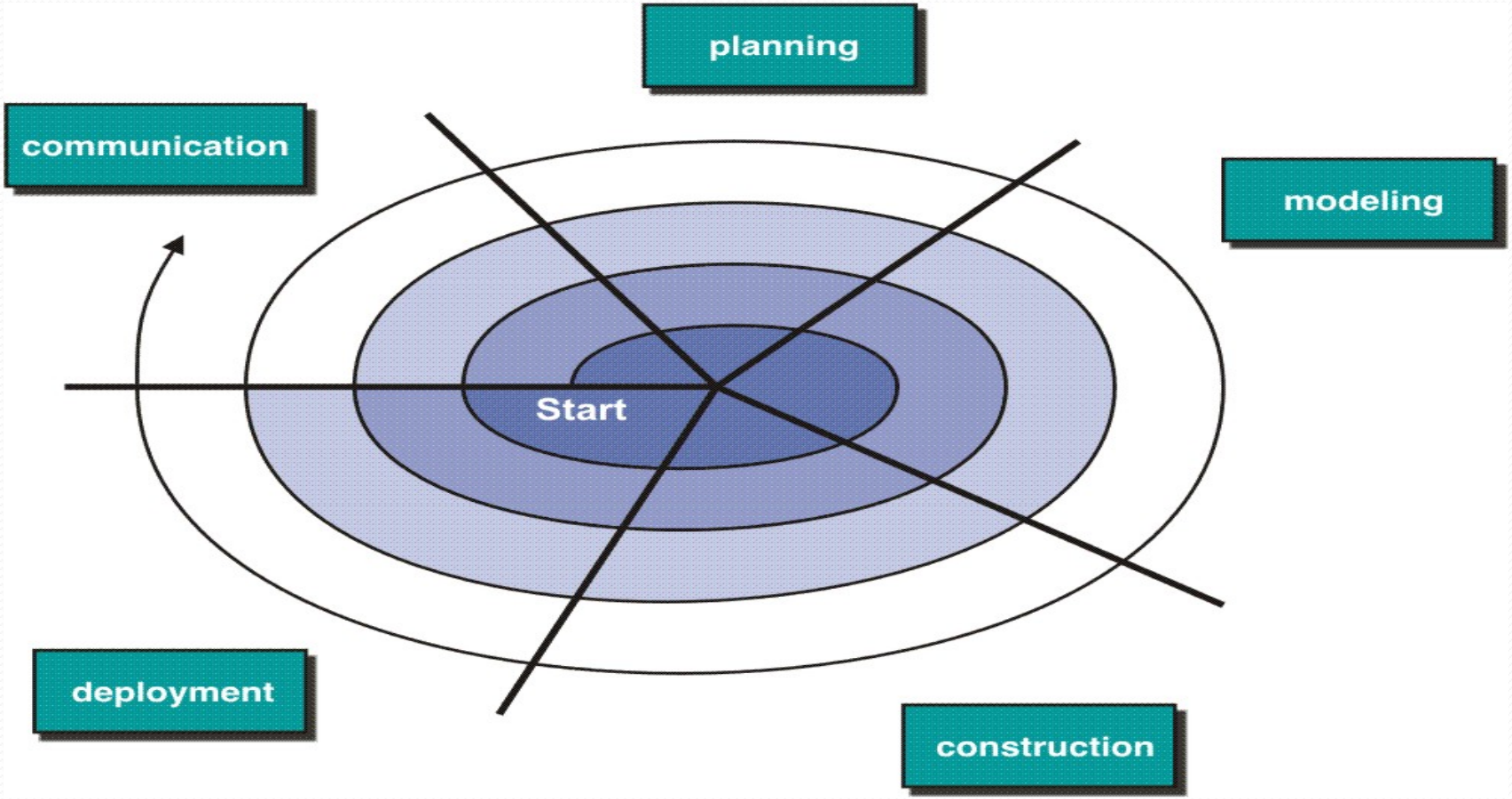
- Prototyping :
  - The **Software Prototyping** refers to building **software** application **prototypes** which displays the functionality of the product under **development**, but may not actually hold the exact logic of the original **software**
  - Developing a prototype based on the stakeholders requirements
  - Unsure of the algorithm implemented.
  - Many systems are built one after the other.
  - Major of them are throwaways



Prototyping

- **Best approach when**:
  - Objectives defined by customer are general but does not have details like input, processing, or output requirement.
  - Developer may be unsure of the efficiency of an algorithm, O.S., or the form that human machine interaction should take.
- It can be used as standalone process model.
- Model assist software engineer and customer to better understand what is to be built when requirements are fuzzy.
- Prototyping start with communication, between a customer and software engineer to define overall objective, identify requirements and make a boundary.

Evolutionary Process Models : Spiral Model



Evolutionary Process Models : Spiral Model

- The *spiral model* is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model.
- It provides potential for rapid development of increasingly more complete version of the software.
- Using spiral, software developed in as series of evolutionary release.
  - Early iteration, release might be on paper or prototype.
  - Later iteration, more complete version of software.
- Evolutionary process begins in a clockwise direction, beginning at the center risk
- Each loop/pass through the spiral model consists of risk assessment and other framework activities from communication through deployment

Evolutionary Process Models : Spiral Model

- First circuit around the spiral might result in development of a product specification.
- Subsequently, develop a prototype and then progressively more sophisticated version of software.
- Unlike other process models that end when software is delivered.
- It can be adapted to apply throughout the life of software.
- It maintains the systematic stepwise approach suggested by the classic life cycle but also incorporates it into an iterative framework activity.
- If risks cannot be resolved, project is immediately terminated

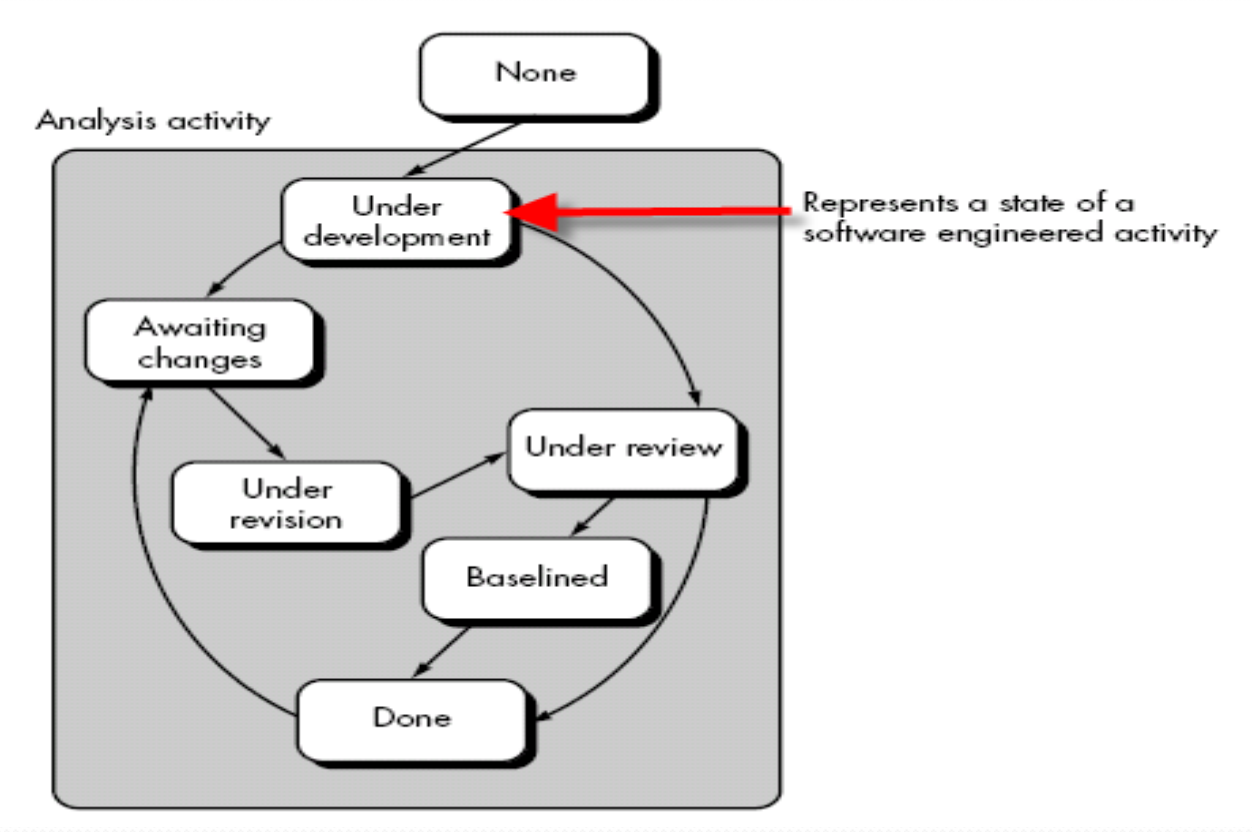
Evolutionary Process Models : Spiral Model

- Disadvantages:
  - Can be a costly model to use.
  - Risk analysis requires highly specific expertise.
  - Project's success is highly dependent on the risk analysis phase.
  - Doesn't work well for smaller projects
  - It may be difficult to convince customers (particularly in contract situations) that the evolutionary approach is controllable.
  - If a major risk is not uncovered and managed, problems will undoubtedly occur.

Evolutionary Process Models : Concurrent Process Models

- The concurrent model is often more appropriate for product engineering projects where different engineering teams are involved.
- In simple words, here, the stages run simultaneously rather than consecutively, with multiple teams working on them.
- They run concurrently but reside in different states

Concurrent Development Model



Concurrent Development

- Visibility of current state of project
- It define network of activities
- Each activities, actions and tasks on the network exists simultaneously with other activities ,actions and tasks.
- Events generated at one point in the process network trigger transitions among the states.

Evolutionary Process Models : Concurrent Process Models

- Disadvantages:
  - Large upfront investment
  - Implementation of early design reviews
  - Dependency on efficient communication
  - Software compatibility and need for computer modeling

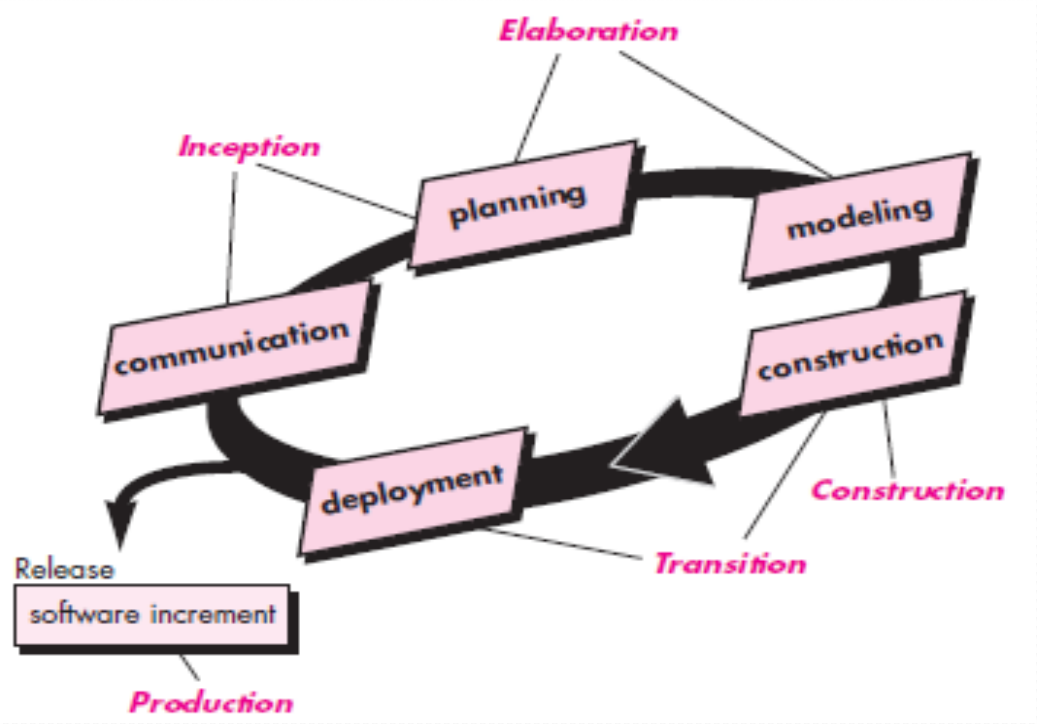
Unified Process

- It can be defined as —object oriented, use case driven, architecture-centric, iterative and incremental software process.
- It is a popular iterative and incremental software development process framework.
- It is a combination of best features of object oriented analysis and design methods.
- This resulted in a robust notation of modeling and design i.e., UML-*Unified Modeling Language*.
- After a refinement of generic processes, its called Rational Unified Process-RUP.
- RUP has online repositories, templates and manuals for development process and management.

Unified Process

- The process is extensible and features include:
  - Iterative Development
  - Requirements Management
  - Component-Based Architectural Vision
  - Visual Modeling of Systems
  - Quality Management
  - Change Control Management

Unified Process



Unified Process

- UP has 5 phases as seen in generic process model
  - Inception Phase
    - Overriding goal is obtained from all interested parties
    - Initial requirements capture
    - Cost Benefit Analysis
    - Initial Risk Analysis
    - Project scope definition
    - Defining a candidate architecture
    - Development of a disposable prototype
    - Initial Use Case Model (10% -20% complete)
    - First pass at a Domain Model

Unified Process

- Elaboration Phase
  - Encompasses communication and modeling activities.
  - Use cases are refined and expanded
  - Uses cases are architecturally represented using
    - Use case model
    - Requirements model
    - Design model
    - Implementation model
    - Deployment model
  - Modifications to plan are often made here.
  - Plan is carefully reviewed.

Unified Process

- Construction Phase
  - Focus is on implementation of the design
  - cumulative increase in functionality
  - greater depth of implementation
  - greater stability begins to appear
  - implement all details, not only those of central architectural value

Unified Process

- Transition Phase
  - Transfer of the system to the user community
  - Includes manufacturing, shipping, installation, training, technical support and maintenance
  - Development team begins to shrink
  - Control is moved to maintenance team
  - Alpha, Beta, and final releases
  - Software updates
  - Integration with existing systems (legacy, existing versions, etc.)

Unified Process

- Production Phase
  - Coincides with the deployment activity.
  - Software is monitored
  - Support for the infrastructure is provided
  - Defect reports and requests for changes are submitted and evaluated.

What is Agility?

- Meaning : able to move quickly and easily
- Effective response to change
- Effective communication among all stakeholders
- Drawing the customer onto the team; eliminate the “us and them” attitude
- Organizing a team so that it is in control of the work performed
- Rapid, incremental delivery of software

Agility

- Agile engineers believe
  - Current software development processes are too heavyweight or cumbersome
  - Too many things are done that are not directly related to software product being produced
  - Current software development is too rigid
  - Difficulty with incomplete or changing requirements
  - Short development cycles (Internet applications)
  - More active customer involvement needed
  - CMM(capability maturity model) focuses on process

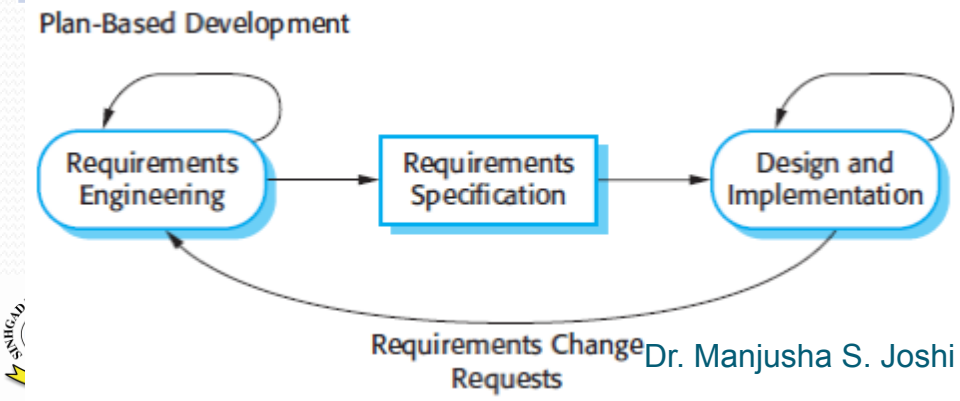
- Agile methods are considered
  - Lightweight
  - People-based rather than Plan-based
  - Set of principles
  - Developed by Agile Alliance—Agile is dynamic, content specific, aggressively change embracing, and growth oriented

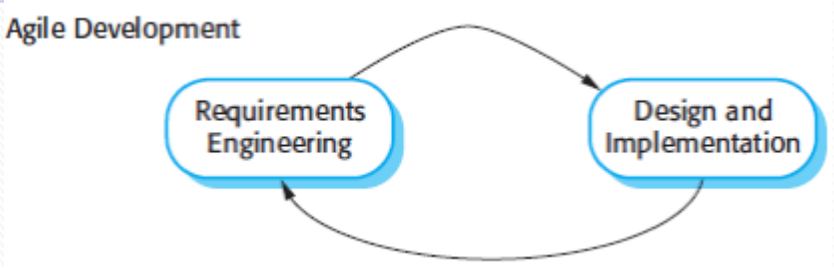
- An Agile Process
  - Is driven by customer descriptions of what is required (scenarios)
  - Recognizes that plans are short-lived
  - Develops software iteratively with a heavy emphasis on construction activities
  - Delivers multiple software increments
  - Adapts as changes occur
- Agile Methods
  - Scrum
  - Extreme Programming
  - Adaptive Software Development (ASD)
  - Dynamic System Development Method (DSDM), etc.
- Agile Alliance (www.agilealliance.org)
  - A non-profit organization promotes agile development

- Agility Principles
  1. Highest priority is to satisfy the customer through early and continuous delivery of valuable software.
  2. Welcome changes from customer at anytime in the process.
  3. Deliver working software frequently
  4. Business people and developers must work together daily throughout the project.
  5. Build projects around motivated individuals.
  6. Have face-to-face conversation.
  7. Working software is the primary measure of progress.
  8. 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
  11. The best architectures, requirements, and designs emerge from self–organizing teams.
  12. At regular intervals, team tune up and adjust behavior accordingly

Plan-driven and agile development

Plan Driven Approach	Agile Approach
Plan Driven Approach identifies separate stages in the software process with outputs associated with each stage	Agile approach consider design and implementation to be the central activities in the software process
Iteration occurs within activities.	Iteration occurs across activities.
Supports incremental development and delivery.	Not code-focused and end up producing a design document





Success in Industry

- The success of projects, which follow Extreme Programming practices, is due to-
  - Rapid development.
  - Immediate responsiveness to the customer's changing requirements.
  - Focus on low defect rates.
  - System returning constant and consistent value to the customer.
  - High customer satisfaction.
  - Reduced costs.
  - Team cohesion and employee satisfaction

Agile Methodologies

- Kanban - Kanban is a simple, visual means of managing projects that emphasizes visibility.
- Scrum - Scrum focuses on breaking a project down into sprints, and only planning and managing one sprint at a time. Scrum also has unique project roles, including a scrum master and Product Owner.
- Extreme Programming - It focuses on continuous development and customer delivery and uses intervals or sprints similar to a scrum methodology

Agile Methodologies

- Feature-Driven Development (FDD) - This methodology involved creating software models every two weeks and requires a development and design plan for every software model feature. Therefore, it has more rigorous documentation requirements than XP. FDD breaks projects down into five basic activities:
    - Develop an overall model
    - Build a feature list
    - Plan by feature
    - Design by feature
    - Build by feature

Agile Methodologies

- Dynamic Systems Development Method (DSDM) -

The Dynamic Systems Development Method (DSDM) cropped up out of the need to provide a common industry framework for rapid software delivery. Part of DSDM is the mandate that rework is to be expected, and any development changes that occur must be reversible.This framework is based on eight key principles:

- Focus on the business need
- Deliver on time
- Collaborate
- Never compromise quality
- Build incrementally from firm foundations
- Develop iteratively
- Communicate continuously and clearly
- Demonstrate control

Agile Methodologies

- Dynamic Systems Development Method (DSDM) -

The Dynamic Systems Development Method (DSDM) cropped up out of the need to provide a common industry framework for rapid software delivery. Part of DSDM is the mandate that rework is to be expected, and any development changes that occur must be reversible.This framework is based on eight key principles:

- Focus on the business need
- Deliver on time
- Collaborate
- Never compromise quality
- Build incrementally from firm foundations
- Develop iteratively
- Communicate continuously and clearly
- Demonstrate control

Introduction to Agile Tools

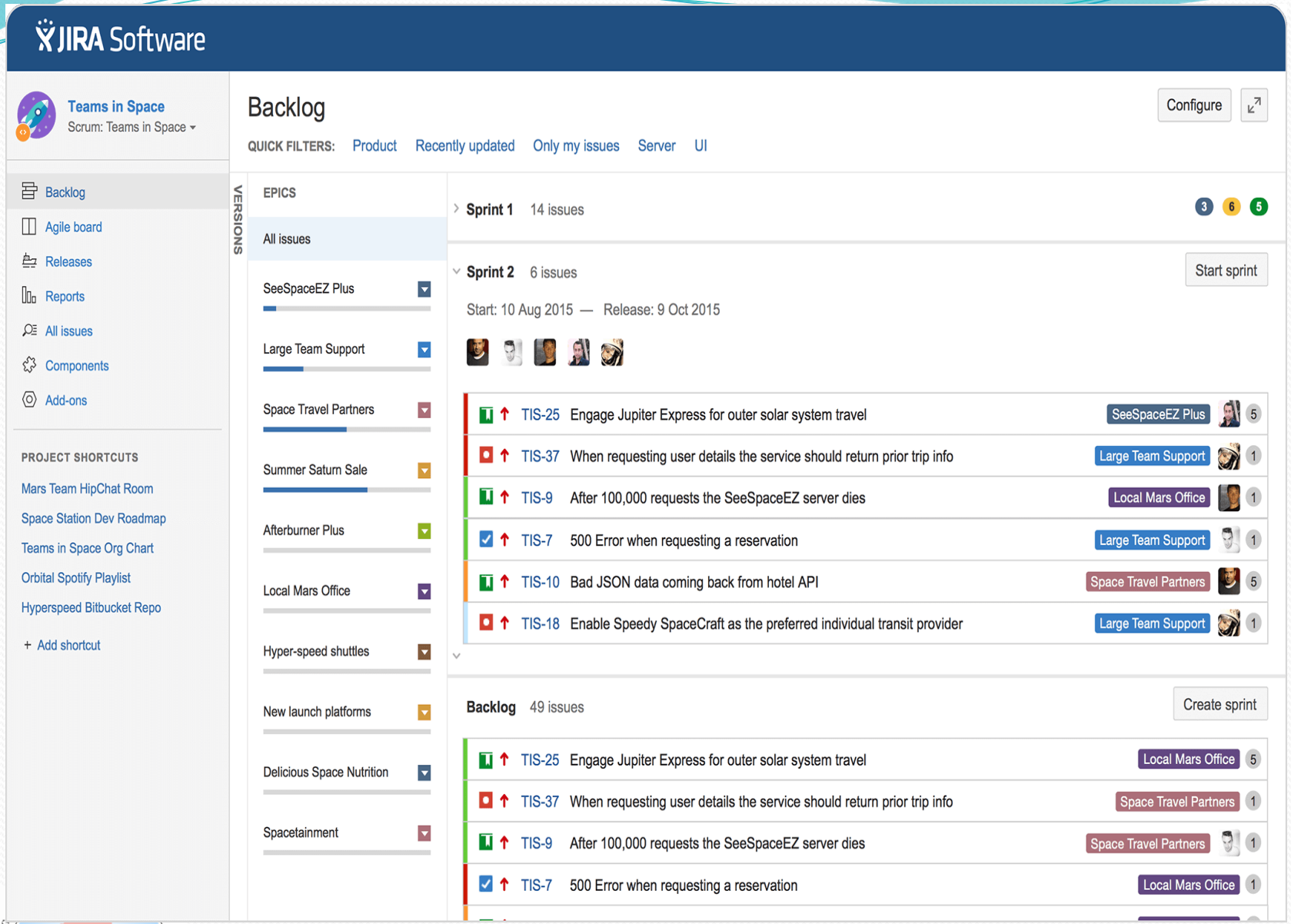
- The Objective is to help in one or more activities in agile development.
- Automated support for project planning, use case development, requirement gathering, code generation and testing.
- Project management tools focus on preparing —*Earned Value Charts*‖ *instead of preparing the Gantt Charts.*
- Purpose is to enhance the environment

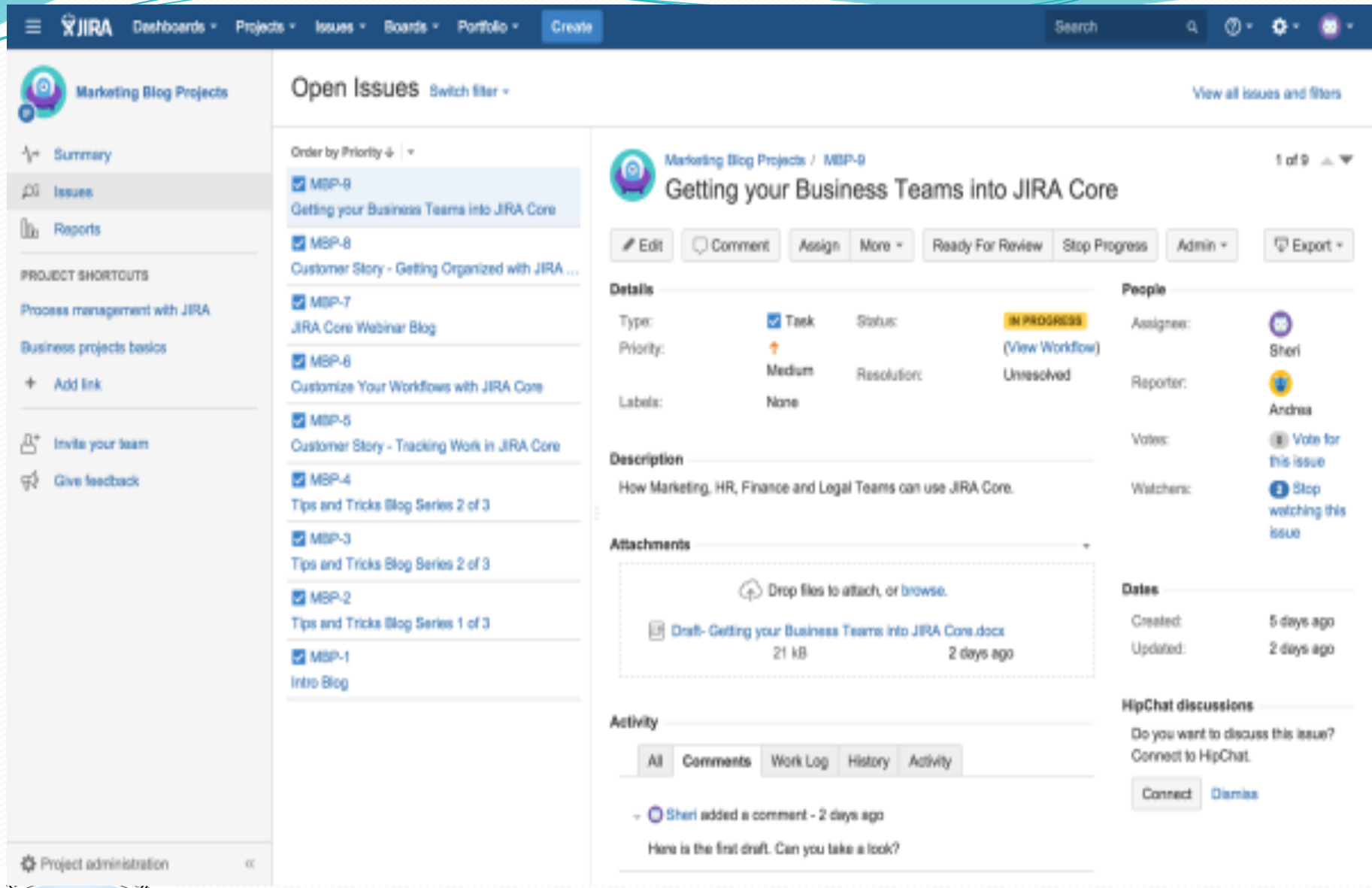
JIRA

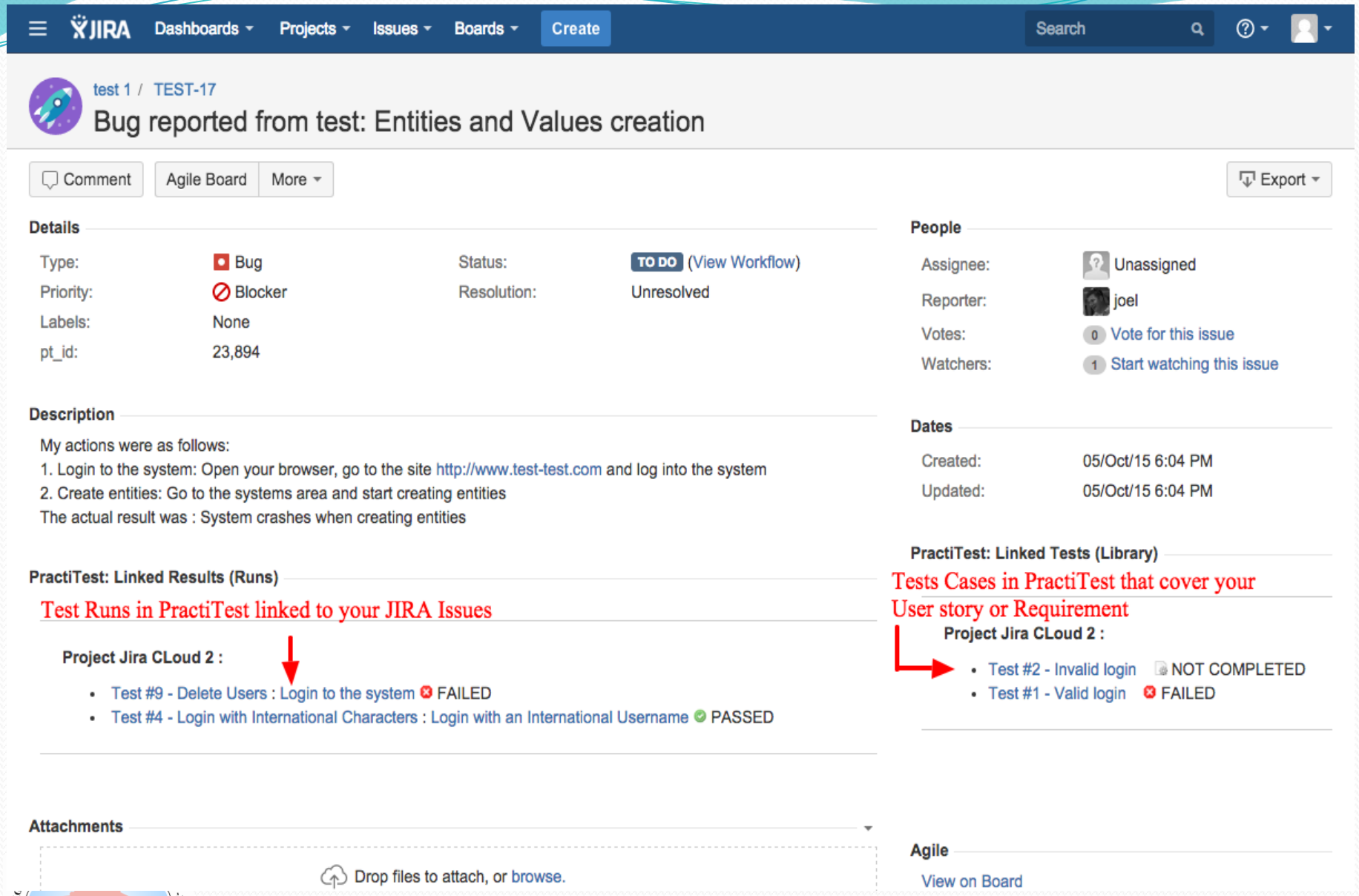
- Developed by Atlassian in 2002.
- Used for bug tracking, issue tracking and project management.
- Inherited from Japanese word —Gojira meaning Godzilla
- JIRA is a competitor for Bugzilla.
- Written in Java and used for Project Management activities.
- Features:
  - Plan development iterations
  - Iteration reports
  - Bug tracking
- JIRA Issues:
  - It is a part of JIRA tool which is used to track bugs in the project.
- JIRA Components
  - Sub-sections of project
  - Used to group issues into smaller parts.
  - Projects can be broken down into features, teams, modules, sub-projects, etc.
  - We can generate reports, collect statistics and display it on dashboards

JIRA

- Reports in JIRA
- Progress of the project can be represented using BurndownChart.
- Different reports like Sprint Report, Version Report, Control Chart, Cumulative flow diagram, etc. are also available.


JIRA Software
Teams in Space
Scrum: Teams in Space
Backlog
Agile board
Releases
Reports
All issues
Components
Add-ons
PROJECT SHORTCUTS
Mars Team HipChat Room
Space Station Dev Roadmap
Teams in Space Org Chart
Orbital Spotify Playlist
Hyperspeed Bitbucket Repo
+ Add shortcut
Backlog
Configure
QUICK FILTERS: Product Recently updated Only my issues Server UI
VERSIONS
EPICS
All issues
SeeSpaceEZ Plus
Large Team Support
Space Travel Partners
Summer Saturn Sale
Afterburner Plus
Local Mars Office
Hyper-speed shuttles
New launch platforms
Delicious Space Nutrition
Spacetainment
Sprint 1 14 issues
Sprint 2 6 issues
Start sprint
Start: 10 Aug 2015 — Release: 9 Oct 2015
TIS-25 Engage Jupiter Express for outer solar system travel SeeSpaceEZ Plus 5
TIS-37 When requesting user details the service should return prior trip info Large Team Support 1
TIS-9 After 100,000 requests the SeeSpaceEZ server dies Local Mars Office 1
TIS-7 500 Error when requesting a reservation Large Team Support 1
TIS-10 Bad JSON data coming back from hotel API Space Travel Partners 5
TIS-18 Enable Speedy SpaceCraft as the preferred individual transit provider Large Team Support 1
Backlog 49 issues
Create sprint
TIS-25 Engage Jupiter Express for outer solar system travel Local Mars Office 5
TIS-37 When requesting user details the service should return prior trip info Space Travel Partners 1
TIS-9 After 100,000 requests the SeeSpaceEZ server dies Space Travel Partners 1
TIS-7 500 Error when requesting a reservation Local Mars Office 1


JIRA
Dashboards
Projects
Issues
Boards
Portfolio
Create
Search
Marketing Blog Projects
Summary
Issues
Reports
PROJECT SHORTCUTS
Process management with JIRA
Business projects basics
Add link
Invite your team
Give feedback
Project administration
Open Issues
Switch filter
View all issues and filters
Order by Priority
MBP-9
Getting your Business Teams into JIRA Core
MBP-8
Customer Story - Getting Organized with JIRA ...
MBP-7
JIRA Core Webinar Blog
MBP-6
Customize Your Workflows with JIRA Core
MBP-5
Customer Story - Tracking Work in JIRA Core
MBP-4
Tips and Tricks Blog Series 2 of 3
MBP-3
Tips and Tricks Blog Series 2 of 3
MBP-2
Tips and Tricks Blog Series 1 of 3
MBP-1
Intro Blog
Marketing Blog Projects / MBP-9
1 of 9
Getting your Business Teams into JIRA Core
Edit
Comment
Assign
More
Ready For Review
Stop Progress
Admin
Export
Details
Type: Task
Priority: Medium
Labels: None
Status: IN PROGRESS
(View Workflow)
Resolution: Unresolved
People
Assignee: Sheri
Reporter: Andrea
Votes: Vote for this issue
Watchers: Stop watching this issue
Description
How Marketing, HR, Finance and Legal Teams can use JIRA Core.
Attachments
Drop files to attach, or browse.
Draft- Getting your Business Teams into JIRA Core.docx
21 kB
2 days ago
Dates
Created: 5 days ago
Updated: 2 days ago
Activity
All
Comments
Work Log
History
Activity
Sheri added a comment - 2 days ago
Here is the first draft. Can you take a look?
HipChat discussions
Do you want to discuss this issue? Connect to HipChat.
Connect
Dismiss


JIRA
Dashboards
Projects
Issues
Boards
Create
Search
test 1 / TEST-17
Bug reported from test: Entities and Values creation
Comment
Agile Board
More
Export
Details
Type: Bug
Priority: Blocker
Labels: None
pt_id: 23,894
Status: TO DO (View Workflow)
Resolution: Unresolved
People
Assignee: Unassigned
Reporter: joel
Votes: 0 Vote for this issue
Watchers: 1 Start watching this issue
Description
My actions were as follows:
1. Login to the system: Open your browser, go to the site http://www.test-test.com and log into the system
2. Create entities: Go to the systems area and start creating entities
The actual result was : System crashes when creating entities
Dates
Created: 05/Oct/15 6:04 PM
Updated: 05/Oct/15 6:04 PM
PractiTest: Linked Results (Runs)
Test Runs in PractiTest linked to your JIRA Issues
Project Jira CLoud 2 :
Test #9 - Delete Users : Login to the system FAILED
Test #4 - Login with International Characters : Login with an International Username PASSED
PractiTest: Linked Tests (Library)
Tests Cases in PractiTest that cover your User story or Requirement
Project Jira CLoud 2 :
Test #2 - Invalid login NOT COMPLETED
Test #1 - Valid login FAILED
Attachments
Drop files to attach, or browse.
Agile
View on Board