Agile Development and Planning

Instructor:
John J Rofrano
Senior Technical Staff Member, DevOps Champion
IBM T.J. Watson Research Center
rofrano@us.ibm.com
@JohnRofrano

Let’s Get Forkin’

- The source for this lab can be FORKED from:

`https://github.com/rofrano/devops-workshop`

You must fork the code so that you can push back to your own copy and trigger the DevOps Pipeline that we will set up

Fork Me On GitHub

Fork me on GitHub

https://github.com/rofrano/devops-workshop

Search or jump to... / Pull requests Issues Marketplace Explore

rofrano / devops-workshop

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

DevOps Workshop Training that uses Vagrant, VirtualBox, Docker, and Minikube

Edit

Manage topics

4 commits 1 branch 0 releases 1 contributor Apache-2.0

Branch: master New pull request Create new file Upload files Find File Clone or download

rofrano Expanded on using vagrant commands Latest commit 123fcce 7 days ago

.gitignore	Initial load	7 days ago
LICENSE	Initial commit	8 days ago
README.md	Expanded on using vagrant commands	7 days ago
Vagrantfile	Initial load	7 days ago

README.md

DevOps Workshop

Fork Me On GitHub

Fork me on GitHub

Fork Me On GitHub

Fork me on GitHub

https://github.com/rofrano/devops-workshop

Search or jump to... / Pull requests Issues Marketplace Explore

rofrano / devops-workshop

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

DevOps Workshop Training that uses Vagrant, VirtualBox, Docker, and Minikube

Edit

Manage topics

4 commits 1 branch ache-2.0

Fork the code to your own account

Branch: master New pull request Create new file Upload files Find File Clone or download

rofrano Expanded on using vagrant commands Latest commit 123fcce 7 days ago

.gitignore	Initial load	7 days ago
LICENSE	Initial commit	8 days ago
README.md	Expanded on using vagrant commands	7 days ago
Vagrantfile	Initial load	7 days ago

README.md

DevOps Workshop

A Forked Repo Example

A Forked Repo Example

A Forked Repo Example

A Forked Repo Example

A Forked Repo Example

A Forked Repo Example

Clone your Fork and Vagrant UP!

- You need to clone your fork of my repo to bring it down to your computer

```
git clone https://github.com/<your_git_account>/devops-workshop.git
cd devops-workshop
vagrant up
```

Objectives

The objectives of this class are to:

- Gain a good overview Agile Development and ZenHub
- How to plan using a Kanban Board and Backlogs
- How to use Epics, Stories, Story Points
- How to create Milestones and use Burndown Charts

“I love deadlines. I like the whooshing sound they make as they fly by.”

–Douglas Adams

How do you avoid this?

Navigating the Unknown

- It would be difficult, if not impossible, to plot a course to the other side of these penguins by this vantage point
  - btw, the penguins will move as you do
- But we can start the journey with what we do know, and as we move forward, the course will reveal itself from our more advanced vantage point
- Developing software is no different

“Don’t decide everything at the point that you know the least”

Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

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

That is, while there is value in the items on
the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

Agile Development

- Requirements and solutions evolve through the collaborative effort of **self-organizing** and **cross-functional** teams and their customers
- It advocates **adaptive planning**, evolutionary development, early delivery, and **continual improvement**
- It encourages rapid and flexible **response to change**

Agile and Scrum

Scrum is the most popular Agile development framework

Agile and Scrum

- **Agile** is a PHILOSOPHY for doing work
  - Not prescriptive
- **Scrum** is a METHODOLOGY for doing work
  - That adds PROCESS to Agile thinking

Scrum

Easy to understand — Difficult to master

- A management framework for incremental product development using one or more small cross-functional, self-organizing teams
- Provides a structure of roles, meetings, rules, and artifacts
- Uses fixed-length iterations, called Sprints, which are typically two weeks long
  - Scrum teams attempt to build a potentially shippable (properly tested) product increment every iteration

Organization of Scrum Teams

- Small team (7 +/- 2)
- Dedicated
- Co-located
- Cross-functional
- Self managing

Copyright © 2012, Kenneth S. Rubin and Innolution, LLC. All Rights Reserved.

Scrum Roles

- Product Owner
- Scrum Master
- Development Team

Copyright © 2012, Kenneth S. Rubin and Innolution, LLC. All Rights Reserved.

Product Owner

- Represents the stakeholder interests
- Responsible for product vision
- Final arbiter of requirements questions
- Constantly re-prioritizes the Product Backlog, adjusting any expectations such as release plans
- Accepts or rejects each product increment
- Decides whether to ship
- Decides whether to continue development
- May contribute as a team member

Scrum Master

(Agile Coach)

- Facilitates the Scrum process
- Creates an environment conducive to team self-organization
- Shields the team from external interference and distractions to keep it "in the zone"
- Helps resolve impediments
- Enforces Sprint timeboxes
- Captures empirical data to adjust forecasts
- Has no management authority over the team (anyone with authority over the team is by definition not its ScrumMaster)

Scrum Team

- Cross-functional (e.g., includes members with testing skills, and often others not traditionally called developers: business analysts, domain experts, etc.)
- Self-organizing / self-managing, without externally assigned roles
- Consists of 5 ± 2 dedicated co-located collaborative members
  - Most successful when located in one team room, particularly for the first few Sprints
  - Most successful with long-term, full-time membership. Scrum moves work to a flexible learning team and avoids moving people or splitting them between teams.
- Negotiates commitments with the Product Owner — one Sprint at a time
- Has autonomy regarding how to reach commitments

Agile Development Is Iterative

Iteration Detail

Agile Planning

DevOps Pipeline

Agile Planning Tools

Pivotal Tracker

Taiga

waffle.io

JIRA

ZenHub.io

ZenHub.io

ZenHub is a Tool for Managing Agile Development

Using ZenHub for Planning

- ZenHub is a plug-in to GitHub
- It provides a Kanban Board and Burndown Charts for project management
- Customizable and integrated with GutHub (one tool)

ZenHub - app.zenhub.com

ZenHub - app.zenhub.com

ZenHub - app.zenhub.com

Kanban Board

What is a KanBan Board?

Kanban Example

Default ZenHub Pipelines

Default ZenHub Pipelines

Default ZenHub Pipelines

Sprint

- A Sprint is one iteration through the design, code, test, deploy cycle
- Usually 2 weeks in duration but could be shorter or longer (but not too much longer... remember working in "small batches")
- Every Sprint should have a Goal this way everyone knows what goal they are working towards

Steps in the Scrum process

The 5 Scrum Meetings

Backlog Refinement

The 5 Scrum Meetings

The 5 Scrum Meetings

The 5 Scrum Meetings

The 5 Scrum Meetings

The 5 Scrum Meetings

The 5 Scrum Meetings

Agile concepts and GitHub

- Sprint → Milestone
- Epics → Epics
- User Stories → GitHub Issues
- Tasks → Markdown Checklist (- [ ])
- Product backlog → Open issues without a Milestone
- Sprint backlog → Issues with a Milestone

User Stories

What Are User Stories?

- A user story represents a small piece of business value that a team can deliver in an iteration
- While traditional requirements (like use cases) try to be as detailed as possible, a user story is defined incrementally, in three stages:
  1. The brief description of the need
  2. The conversations that happen during backlog grooming and iteration planning to solidify the details
  3. The tests that confirm the story's satisfactory completion

Story driven development

- User Stories document a persona requesting a function to achieve a goal
- The typical form is as follows:
  - **As a** <*some role*>
  - **I need** <*some function*>
  - **So that** <*i get some benefit*>
- User Stories can be entered into GitHub as **Issues**

Defining Done

- I like to include the following in each User Story
- **Assumptions**
  - List any assumptions about how to implement the Story
- **Acceptance Criteria**
  - Define what it means to be "*done*"

Example User Story

Code | Issues 13 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs | Settings

Make the hit counter persistently survive service restarts #4

Edit | New issue

Open **rofrano** opened this issue 2 hours ago · 0 comments

rofrano commented 2 hours ago — Owner

As a User
I need the hit counter to persist the last known count
So that I don't loose track of the count after the service is restarted

Assumptions:

- We will use Redis as the persistent store
- A Redis service from Bluemix should be used

Acceptance Criteria:

```
When I advance the hit counter to 2
And I restart the hit counter services
And I call the hit counter URL
Then I should see 3 returned from the service
```

Projects
None yet

Labels
None yet

Milestone
No milestone

Assignees
No one—assign yourself

1 participant

Make the hit counter persistently survive service restarts

As a User
I need the hit counter to persist the last known count
So that I don't loose track of the count after the service is restarted

Assumptions:

- We will use Redis as the persistent store
- A Redis service from IBM Cloud should be used

Acceptance Criteria:

```
When I advance the hit counter to 2
And  I restart the hit counter services
And  I call the hit counter URL
Then I should see 3 returned from the service
```

Backlog Refinement

Mechanics of Backlog Refinement

- Goal: At the end of Backlog Refinement the New Issues column is empty
- Take Stories from New Issues and...
  - Move them into the ranked Backlog if they will be worked on soon
  - Move them into the Ice Box if they are a good idea but not now
  - Reject them if they are not where you want to go
- Groom the Backlog by ranking the Stories in order of importance and making sure the story contains enough information for a developer to start working on it

Technical Debt

- Technical debt is anything you need to do that doesn't involve creating a new feature
- Technical debt builds up when you take shortcuts
- Examples of Technical Debt:
  - Code refactoring
  - Setup and maintenance of environments
  - Changing technology like databases

Accept or Reject Open Issues

Accept or Reject Open Issues

Sprint Planning

Sprint Planning

Attendees: Product Owner, Scrum Master, Development Team

- The Product Owner is responsible for declaring which Product Backlog Items are the most important to the business during Backlog Refinement
- The Development Team looks at the latest product Increment, projected velocity, and past performance. Based on the data, the Development Team forecast what can be achieved.
- The team Assigns work from the Product Backlog to the Sprint Milestone.
- Plan to spend 4 hours for a 2 week sprint

Sprint Planning

- Steps for creating a Sprint Plan:

  1. Create a Milestone for the Sprint
  2. Assign Issues from the Backlog to the Milestone
  3. Filter ZenHub by the Milestone to work the plan

Sprint Planning Goals

- Each Sprint should have a clearly defined Business Goal
- The Product Owner comes with a proposal of the Sprint Goal and Product Backlog Items supporting it.
- The whole Scrum Team collaborates on crafting the Sprint Goal, so everybody understands why we are building the Increment.

Create a Milestone

Issues · DevOpsResearch/lab-z

Secure https://github.ibm.com/DevOpsResearch/lab-zenhub/issues

This repository Search Pull requests Issues Gist ToDo

DevOpsResearch / lab-zenhub

Watch 1 Star 1 Fork 0

Code Issues 14 Pull requests 0 Boards Reports Projects 0 Wiki

Filters is:issue is:open Labels Milestones New issue

14 Open 5 Closed Author Labels Milestones Assignee Sort

Get the basic service work Epic
#22 opened 18 days ago by rofrano New Issues

Set up a DevOps Pipeline for continuous deployment into Bluemix
#13 opened on Jul 27, 2016 by rofrano New Issues

Need Swagger API documentation
#12 opened on Jul 27, 2016 by rofrano New Issues

Need the ability to remove a counter
#11 opened on Jul 27, 2016 by rofrano New Issues

Need the ability to reset the counter
#10 opened on Jul 27, 2016 by rofrano New Issues

Add the ability to use multiple counters

Create a Milestone

Create a Milestone

Select New Milestone

Select New Milestone

Select New Milestone

New Milestone - DevOpsResea

Secure https://github.ibm.com/DevOpsResearch/lab-zenhub/mile...

DevOpsResearch / lab-zenhub

Watch 1 Star 1 Fork 0

Code | Issues 14 | Pull requests 0 | Boards | Reports | Projects 0 | Wiki

New milestone

Create a new milestone to help organize your issues and pull requests. Learn more about milestones and issues.

Title

Sprint 1

Description

The goal of this sprint is to get a skeletal service running in Bluemix

Due Date (optional) clear

April 2017

Mon	Tue	Wed	Thu	Fri	Sat	Sun
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

Create milestone

Select New Milestone

Select New Milestone

Select New Milestone

Select New Milestone

Mechanics of Sprint Planning

- Take Issues from the top of the ranked Backlog and assign them to the Sprint Milestone
- Groom the Story by clarifying and assigning Story Points, Labels, and making sure the story contain enough information for a developer to start working on it
- Stop adding Stories when your team’s Velocity is reached

Team Velocity

- The number of Story Points a team can complete in a single Sprint
- This will change over time as the team gets better at estimating and better at executing
- The Velocity is unique to the team because the story point assignment is unique to the team

Story Points

- Story Points acknowledge the fact that sizing using absolute time-to-complete is highly inaccurate so it uses an estimate of complexity
- It is measurement of a feature’s size *relative* to other features usually expressed in T-Shirt sized (S, M, L, X) or Fibonacci numbers (1, 2, 3, 5, 8, 13, 21)
- The important thing is to agree on "average" and evaluate from that (i.e., is it the same, larger, or smaller than average)

What Does a Story point measure?

Effort

Complexity

Uncertainty

What Size Should A User Story Be?

- A Story should be small enough to be coded and tested within a single Sprint iteration — ideally just a few days
- When a Story is too large in scope it is considered to be an Epic
- Backlog items tend to start as Epics when they are lower priority and less defined
- For sprint planning, Epics should be broken down into smaller stories, but not so small that you have moved into detailed design.

How Tall Are These Buildings?

This is sometimes called "Planning Poker"

How Tall Are These Buildings?

This is sometimes called "Planning Poker"

How Tall Are These Buildings?

This is sometimes called "Planning Poker"

How Tall Are These Buildings?

This is sometimes called "Planning Poker"

How Tall Are These Buildings?

This is sometimes called "Planning Poker"

What about really big Ideas?

Epics

- Epics are Stories that are so big, they need to be broken up into smaller Stories
  - A single Story should be smaller than a Sprint
  - Epics are usually larger than a Sprint
- Epics are a way of grouping Stories with a common goal together
- Epics can be *larger* than a Milestone

Let’s Make an Epic

Let’s Make an Epic

Daily Execution

Daily Standup

Attendees: Scrum Master, Development Team, Product Owner (optional)

- Occurs every day at the same time and place
- Called a "standup" because everyone should remain standing during the meeting to keep it short
  - Timeboxed to 15 minutes
  - Not a project status meeting — all status should be tabled for later discussion
- Each team member briefly reports on their work

Daily Standup Questions

Attendees: Scrum Master, Development Team, Product Owner (optional)

- Each team member answers three questions:
  1. What did I accomplish the previous day?
  2. What will I work on today?
  3. What blockers or impediments are in my way?

Daily Execution

- Take the next highest priority item from the Sprint Backlog
- Assign it to yourself
- Start working on it
- No one should have more than one story assigned to them unless they are blocked and want to start a second story while waiting
- When you are finished, move the Story to the Done column

Daily Execution

Daily Execution

Measuring Progress

Milestones and Burn-downs

- Milestones can be created for your project
  - e.g., Sprint, Beta Drop, Demo, Release 1, etc.
- Burn-down charts can be used to measure your progress against a Milestone

Burn-Down

- The measurement of Story Points completed vs Story Points remaining for a given Sprint
- Over time the Story Points remaining should go down, hence the name: Burn-down.

Burndown Chart

Sprint Review / Playback

Sprint Review / Playback

Attendees: Product Owner, Scrum Master, Development Team, (optionally Stakeholders + Customers)

- Live Demonstration of implemented Stories
- Product Owner determines if done based on acceptance criteria and those stories are Closed
- Feedback gets converted into new Product Backlog Stories
  - This is where iterative development allows the creation of products that couldn't have been specified up front in a plan-driven approach

Sprint Retrospective

Attendees: Scrum Master, Development Team

- Team reflects on their progress for the Sprint
  - What went well? (keep doing)
  - When did not go well? (stop doing)
  - What should we change for the next Sprint?
- This is *critical* for maintaining a healthy team

Metrics and Feedback

Vanity Metrics

...good for feeling awesome, bad for action

- Consider the total number of daily “hits” to your website is 10,000
- Now what? (what does a "hit" represent?)
  - Do you really know what actions you took in the past that drove those visitors to you?
  - Do you really know which actions to take next?
  - In most cases, I don’t think it’s very helpful

http://tim.blog/2009/05/19/vanity-metrics-vs-actionable-metrics/

Actionable Metrics

- Reduce time-to-market for new features.
- Increase overall availability of the product.
- Reduce the time it takes to deploy a software release.
- Increase the percentage of defects detected in testing before production release.
- Make more efficient use of hardware infrastructure.
- Provide performance and user feedback to the product manager in a more timely manner.

What did we learn?

- Your should have a good overview Agile Development and ZenHub
- How to do planning using a Kanban Board
- How write User Stories and Epics
- How to create Milestones and use Burndown Charts

Additional Reading

- **Scrum Training Series**
  - http://scrumtrainingseries.com
- **Product Owner**
  - https://www.youtube.com/watch?v=502ILHjX9EE
  - https://dzone.com/articles/hiring-42-scrum-product-owner-interview-questions
- **Metrics**
  - https://dzone.com/articles/agile-metricsthe-good-the-bad-and-the-ugly
- **Spotify Engineering Culture**
  - https://labs.spotify.com/2014/03/27/spotify-engineering-culture-part-1/
  - https://labs.spotify.com/2014/09/20/spotify-engineering-culture-part-2/