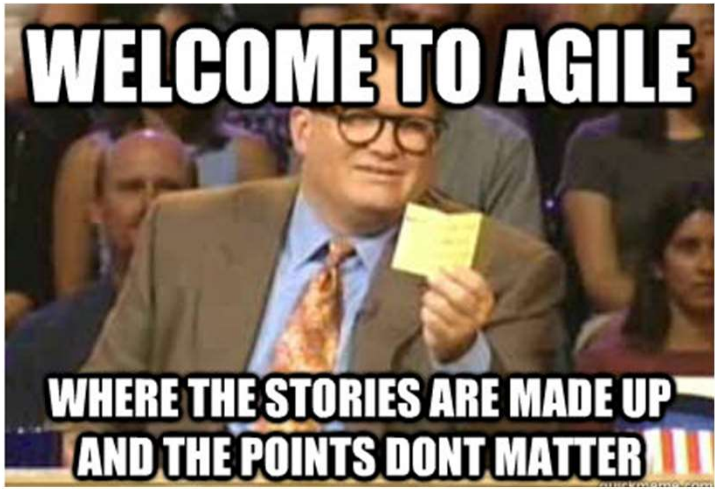
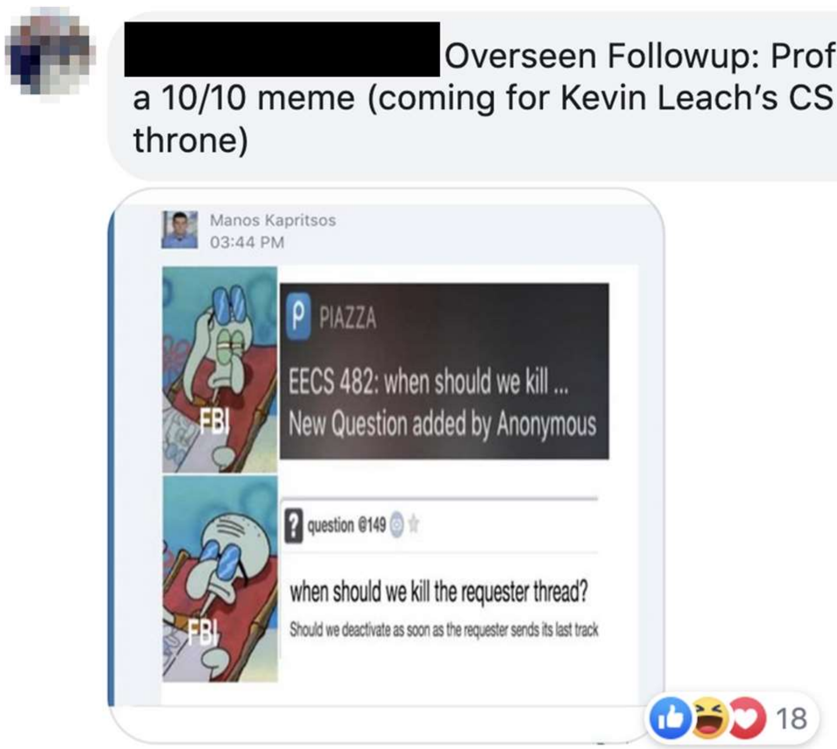
Lecture 5:
Software Engineering Process



1/27/2020

It begins…



[redacted] Overseen Followup: Professor Kapritsos with a 10/10 meme (coming for Kevin Leach's CS professor meme throne)

Manos Kapritsos
03:44 PM

PIAZZA

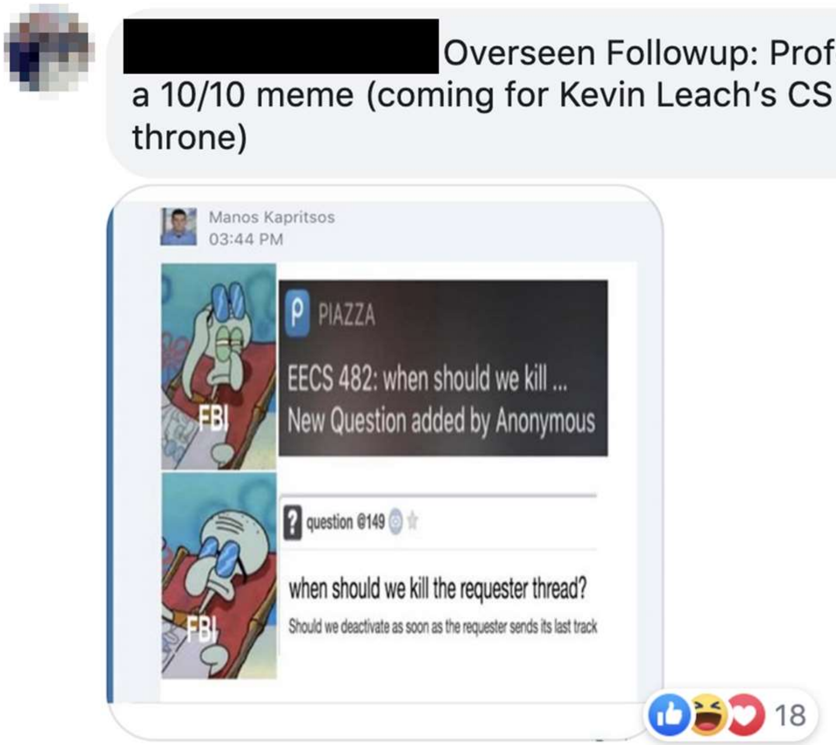
EECS 482: when should we kill ...
New Question added by Anonymous

question @149

when should we kill the requester thread?

Should we deactivate as soon as the requester sends its last track

It begins…



[redacted] Overseen Followup: Professor Kapritsos with a 10/10 meme (coming for Kevin Leach's CS professor meme throne)

Manos Kapritsos
03:44 PM

PIAZZA

EECS 482: when should we kill ...
New Question added by Anonymous

question @149

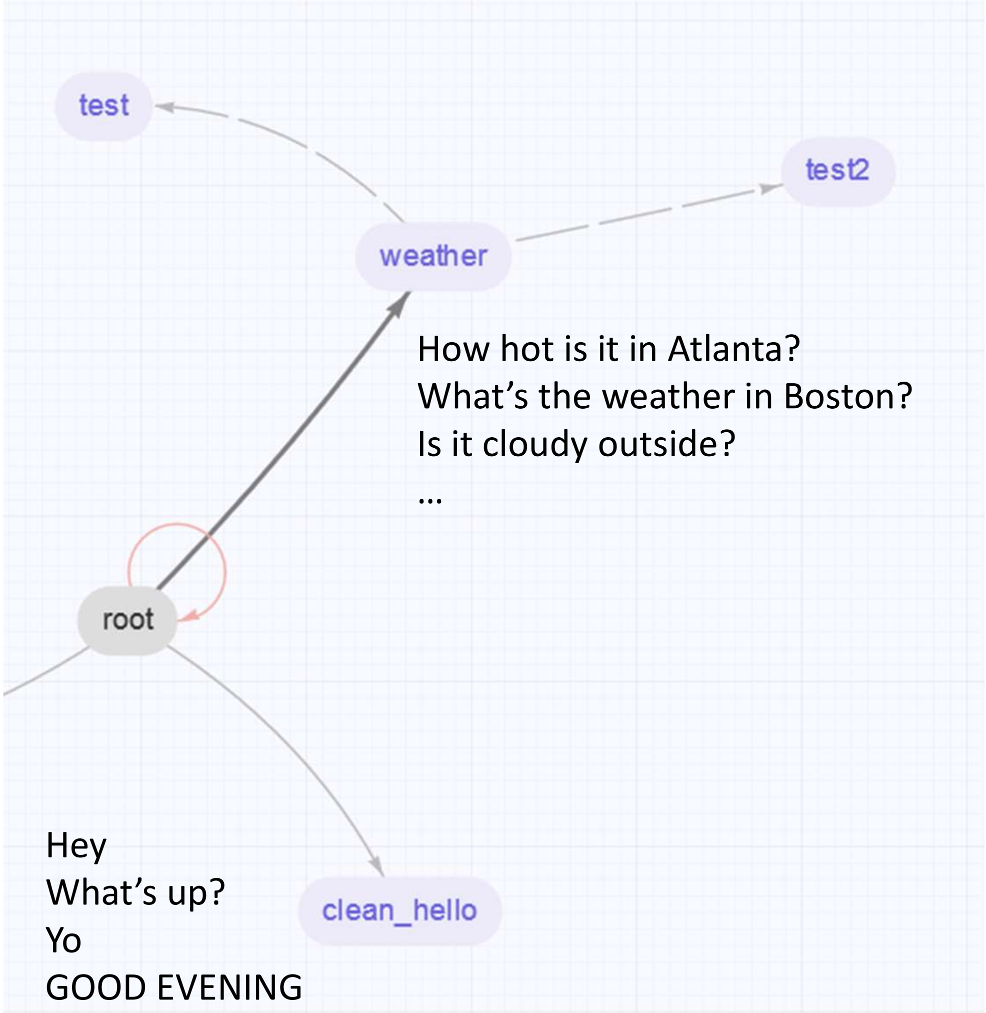
when should we kill the requester thread?

Should we deactivate as soon as the requester sends its last track

A worse ending than Game of Thrones

Review: Competencies

- You used the platform to construct an *informational competency*
  - You saw how **states** are built within competencies, and **transitions** are given example data to help with **intent classification**
  - More examples help strengthen classification power between intents

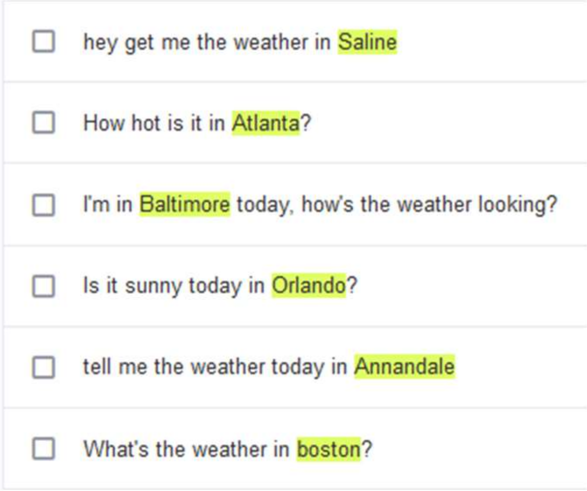


How hot is it in Atlanta?
What’s the weather in Boston?
Is it cloudy outside?
…

Hey
What’s up?
Yo
GOOD EVENING

Review: Slots

- You saw how to label slot information in utterances for each intent

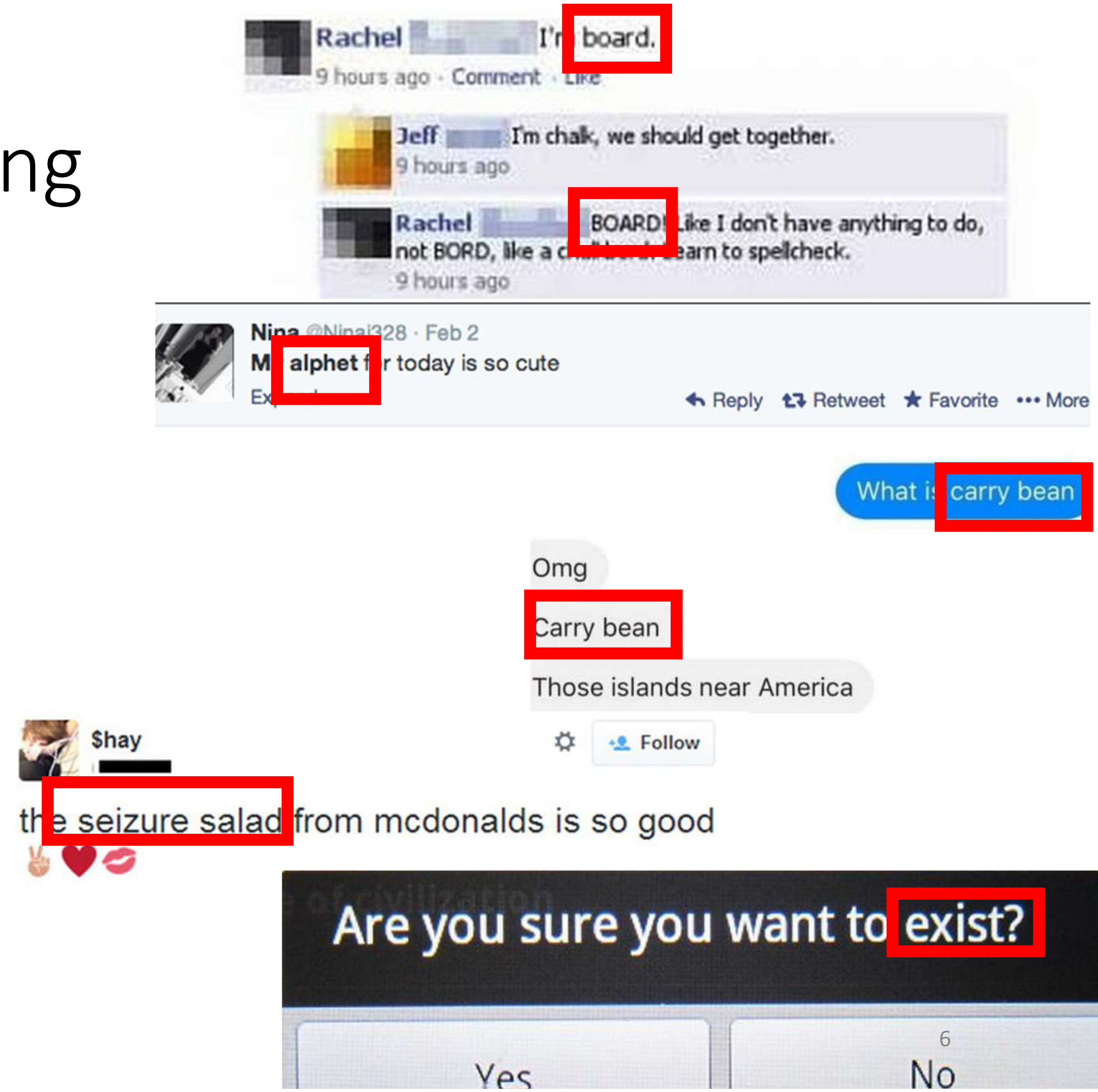


- □ hey get me the weather in Saline
- □ How hot is it in Atlanta?
- □ I'm in Baltimore today, how's the weather looking?
- □ Is it sunny today in Orlando?
- □ tell me the weather today in Annandale
- □ What's the weather in boston?

Okay, how do we make this platform more popular?
Moar colors for slot labels
No autosave!
Less repetitive example entry?
memecenter.com

Review: Slot Mapping

- You build a simple slot mapper to correct for minor misspellings
  - Misspellings frequent for written chat interactions
- Slot mapping still useful for spoken utterances
  - Brand names (e.g., RAV4/Pilot/Expedition → SUV)
  - Nouns where ASR fails (e.g., “EECS” transcribed to “eeks”)



Rachel [redacted] I'm board.
9 hours ago · Comment · Like

Jeff [redacted] I'm chalk, we should get together.
9 hours ago

Rachel [redacted] BOARD! Like I don't have anything to do, not BORD, like a child. Learn to spellcheck.
9 hours ago

Nina @Ninai328 · Feb 2
My alphet for today is so cute
Expand

What is carry bean

Omg

Carry bean

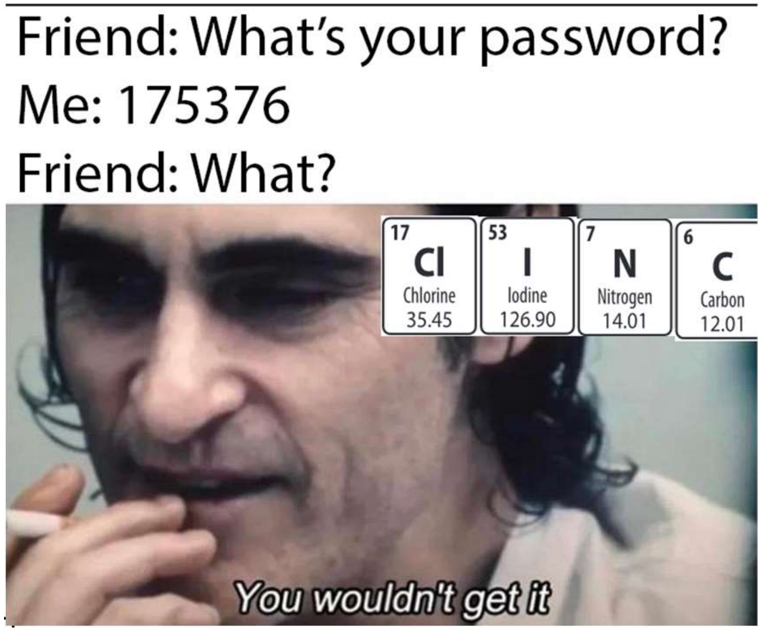
Those islands near America

$hay [redacted]

the seizure salad from mcdonalds is so good ✌❤💋

Are you sure you want to exist?

Clinc platform credentials



- Username: Your `uniqname`
- Password: `pass_uniqname`
- Example: `kjleach` / `pass_kjleach`
- Your institution is `w20_team`*`X`*, where X is your team number
- (you should change your password)

Project Pitches (to present Wednesday!)

- Students on the waitlist:
  - If you **get off the waitlist**, but **aren’t** in a group, please let us know ASAP
    - We will make alternate arrangements
- Example slides online—but feel free to be as creative as you like
- You get **5 minutes exactly** (10 groups in 75 minutes!)
  - We will relay feedback to you after class
  - You will provide feedback to others in writing
- We’ll work with you on scoping later
  - You will schedule time with Brian or Oliver (plus an IA) to help ensure your group project is scoped appropriately

One Slide Summary: Project Pitches and Software Process

- A **pitch** is a brief statement meant to **convince** a **stakeholder** to invest in your idea
  - A successful pitch is **integral** to early investment (e.g., angel investors)
  - Pitching ideas is **pervasive** throughout one’s career
    - Convince a **manager** to allocate resources to your idea
    - Convince **group members** or **subordinates** to work on something
    - Convince a **customer** that your solution is best
- **Software Process** refers to techniques for managing **groups** of software developers
  - Historically, companies have tried many different processes: waterfall, agile, feature-driven, scrum, extreme programming

Project Pitches

- In this class, you will convince an instructor that conversational AI can solve an interesting problem of your choosing
- The biggest challenge is **time**: how do you fit a big idea into a 5-minute pitch?
  - Rehearsal
  - Time preparing visual aids (e.g., think hard what goes in your slides)

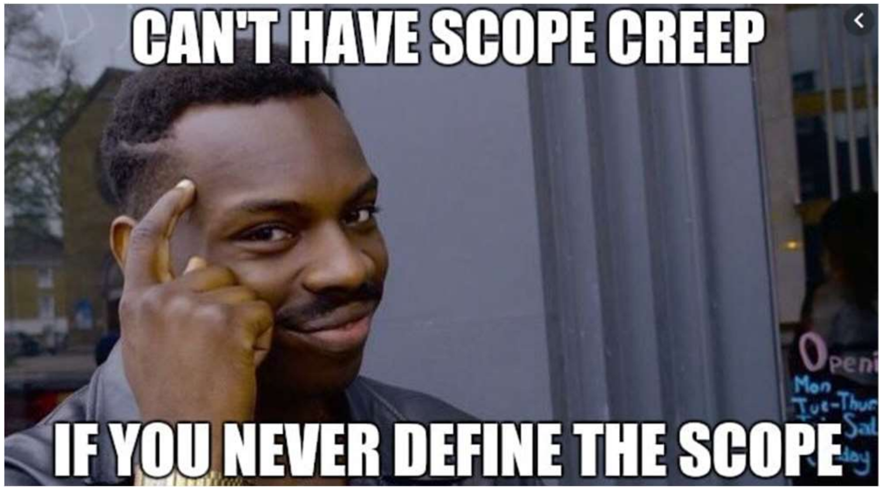
Hey, I just met you, and this is crazy,
but here's my sales pitch,
so buy it, maybe?

What we’re looking for in your pitch

- “Material”: You’ve thought about:
  - a problem where Conversational AI can work
  - a big idea that sounds like it could have high impact
  - an appropriate scope for a semester
  - Pieces that you’ll need to integrate to get things working
    - Third-party APIs, tasks to do
- “Non-technical”:
  - You have a well-done, rehearsed 5-minute pitch
  - Appropriate, visually-helpful slides (e.g., slide numbers)
  - Well-defined objectives/metrics

After the Pitch: Scoping

- Once you’ve settled on an idea, we’ll work with you to help make sure the project is feasible
  - Brian/Oliver will hold scheduled “office hours” where you will meet with them for an hour
  - Your project may change slightly based on this scoping activity



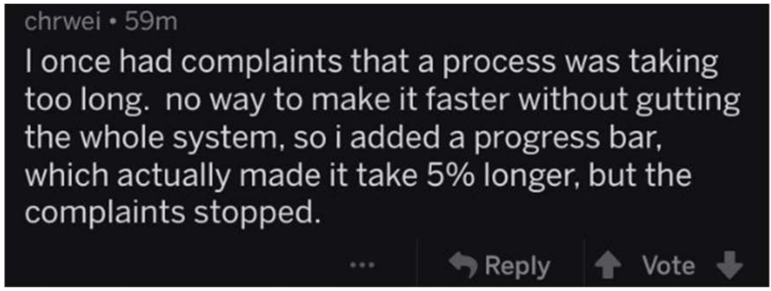
Sprints

- The course is structured around “Sprint Reviews”
  - You will give updates every couple weeks
- **Why?**
  - Agile is a widely-used process among current software and IT companies
  - We want you to develop experience with “agile-like” practices
- **What?**
  - Agile development methodology is a **process** for risk mitigation, planning, and allocation of engineering resources

Process

- **Process** is “the set of activities and associated results that produce a software product.”
- Examples include the waterfall model, spiral development, agile development and extreme programming.

- Naïve example:
  - Discuss objectives
  - Write code
  - Test code
  - Debug code
  - Repeat until done

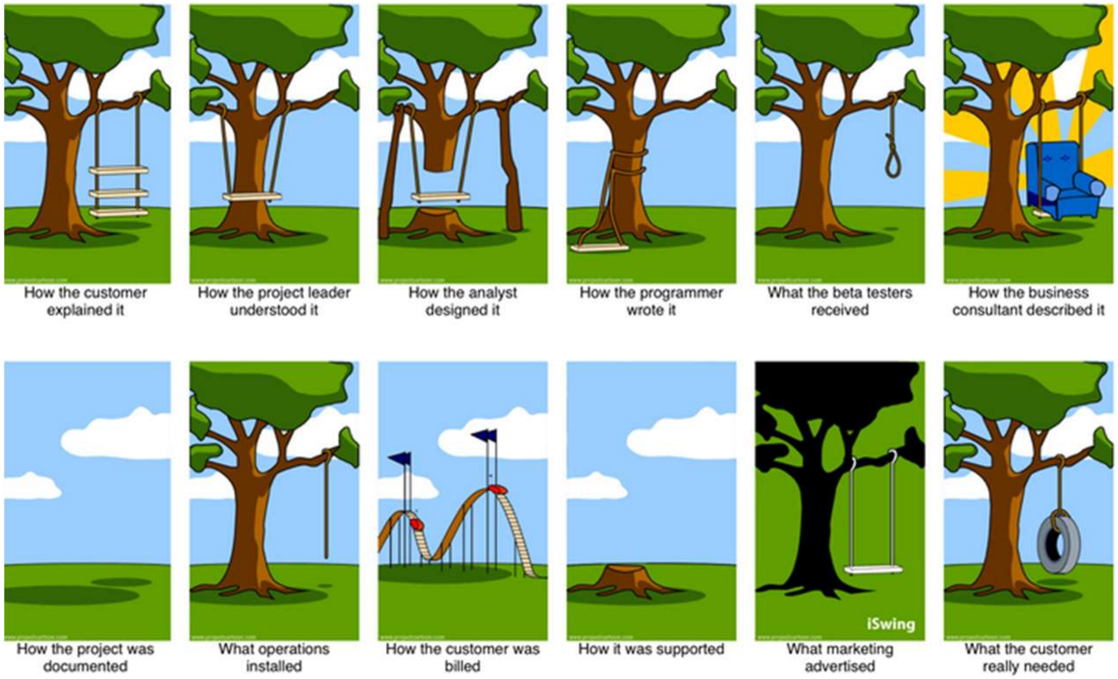


chrwei • 59m

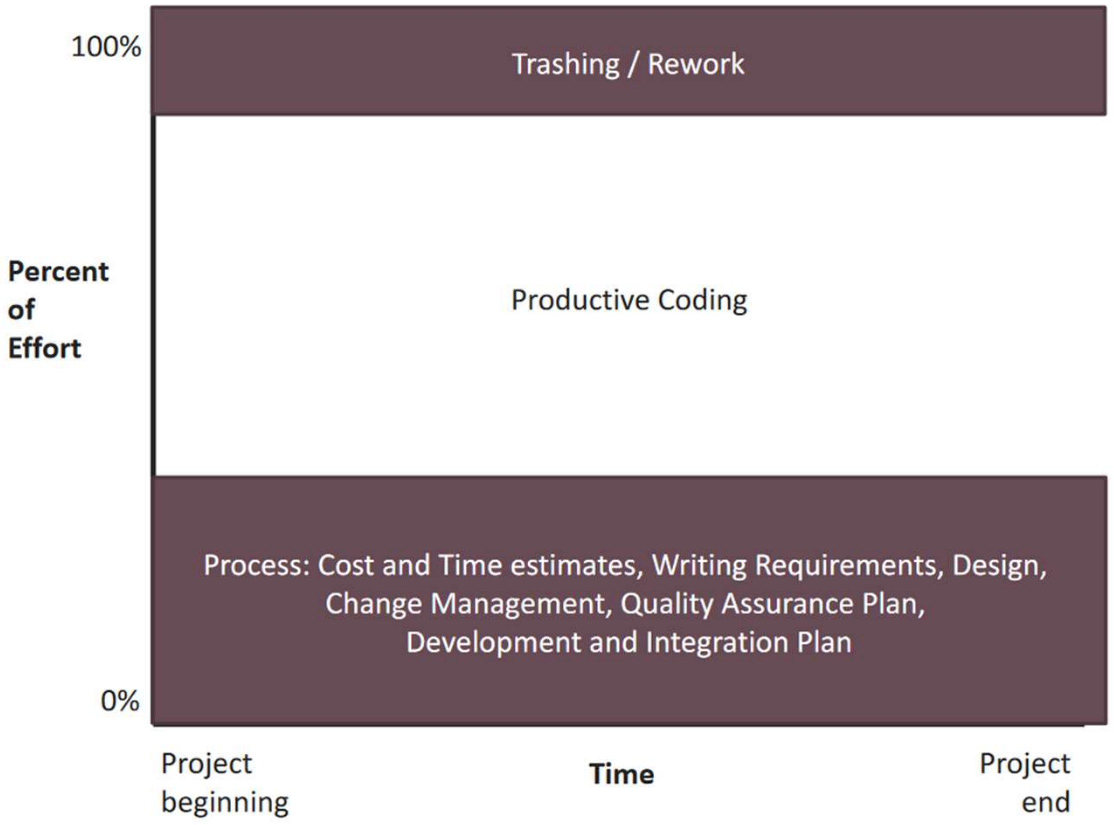
I once had complaints that a process was taking too long. no way to make it faster without gutting the whole system, so i added a progress bar, which actually made it take 5% longer, but the complaints stopped.

Waterfall

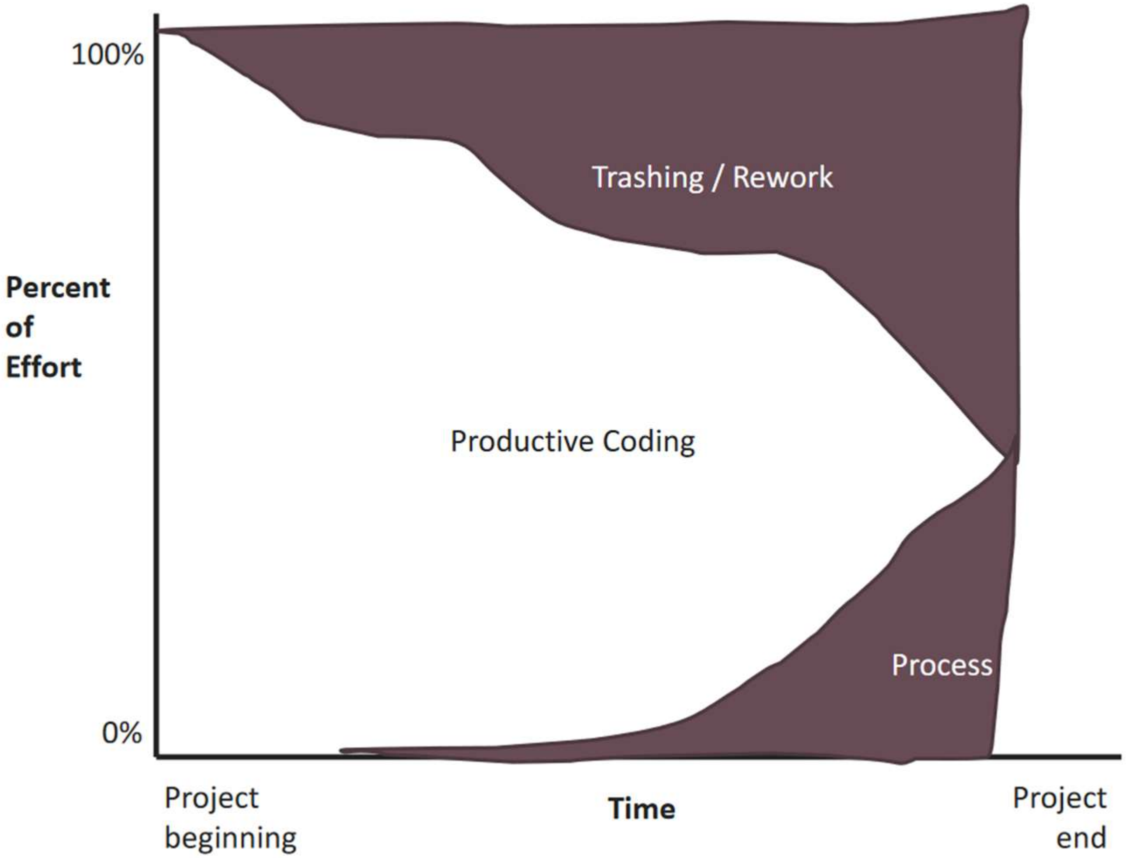
- Requirements Elicitation: pretend to know what the customer wants
- Analysis: ensure the customer can provide enough $
- Design: plan out resource allocation
- Implementation: write a lot of code
- Testing: write a lot of tests
- Maintenance: change your code that didn't capture what the customer wanted


How the customer explained it
How the project leader understood it
How the analyst designed it
How the programmer wrote it
What the beta testers received
How the business consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
iSwing
What marketing advertised
What the customer really needed

Idealized Software Development


100%
Trashing / Rework
Percent
of
Effort
Productive Coding
Process: Cost and Time estimates, Writing Requirements, Design, Change Management, Quality Assurance Plan, Development and Integration Plan
0%
Project beginning
Time
Project end

Reality Without Good Planning



Why Do We Need Process?

- **Requirements**: Mid-project informal agreement to changes suggested by customer or manager.
  - Project scope expands 25-50%
- **Quality Assurance**: Late detection of requirements and design issues. Test-debug-reimplement cycle limits development of new features.
  - Release with known defects
- **Defect Tracking**: Bug reports collected informally.
  - Bugs forgotten
- **System Integration**: Integration of independently developed components at the very end of the project.
  - Interfaces out of sync
- **Source Code Control**: Accidentally overwritten changes.
  - Lost work
- **Scheduling**: When project is behind, developers are asked weekly for new estimates.
  - Project falls further behind

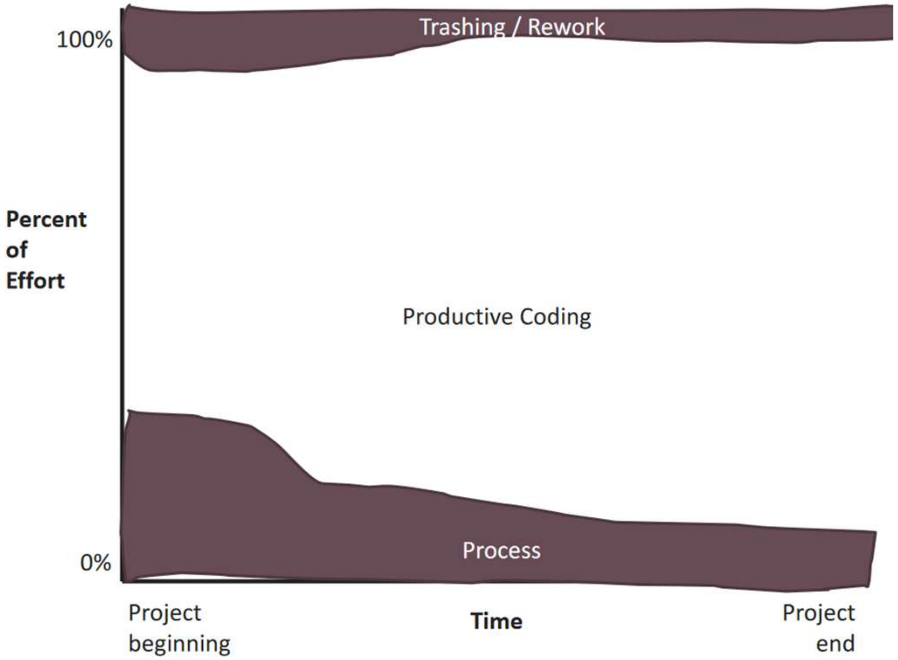
Failures in Process: Survival Mode

- Missed deadlines → “solo development mode”,
  developers stop interacting with testers, technical writers, managers, etc.
- “The producers even set a deadline; they gave a specific date for the end of the crunch, which was still months away from the title's shipping date, so it seemed safe. That date came and went. And went, and went. When the next news came it was not about a reprieve; it was another acceleration: twelve hours six days a week, 9am to 10pm.

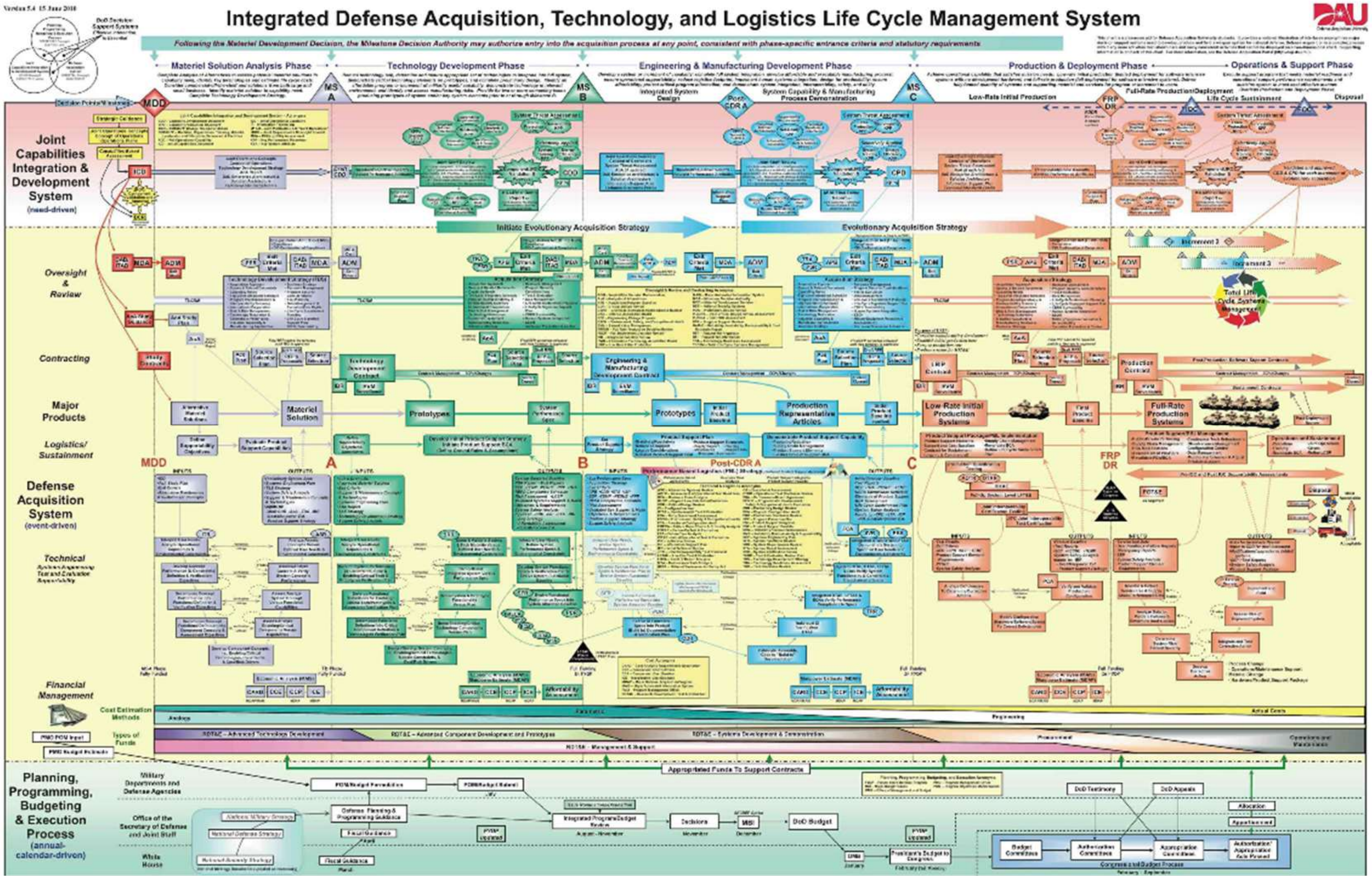
  Weeks passed. Again the producers had given a termination date on this crunch that again they failed. Throughout this period the project remained on schedule. **The long hours started to take its toll on the team**; people grew irritable and some started to get ill. People dropped out in droves for a couple of days at a time, but then the team seemed to reach equilibrium again and they plowed ahead. The managers stopped even talking about a day when the hours would go back to normal.” – EA: The Human Story

Process: Key Idea

- Software Process represents an **up-front time investment**
  - Spend time maintaining process to save time later



Process: Potentially Complicated


Version 5.4 15 June 2010
Integrated Defense Acquisition, Technology, and Logistics Life Cycle Management System
DAU
Following the Materiel Development Decision, the Milestone Decision Authority may authorize entry into the acquisition process at any point, consistent with phase-specific entrance criteria and statutory requirements
Materiel Solution Analysis Phase
Technology Development Phase
Engineering & Manufacturing Development Phase
Production & Deployment Phase
Operations & Support Phase
MDD
MS A
MS B
MS C
Post-CDR A
Integrated System Design
System Capability & Manufacturing Process Demonstration
Low-Rate Initial Production
FRP DR
Full-Rate Production/Deployment
Life Cycle Sustainment
Disposal
IOC
FOC
Joint Capabilities Integration & Development System
(need-driven)
ICD
CDD
CPD
Initiate Evolutionary Acquisition Strategy
Evolutionary Acquisition Strategy
Increment 2
Increment 3
Oversight & Review
Total Life Cycle Systems Management
Contracting
Technology Development Contract
Engineering & Manufacturing Development Contract
LRIP Contract
Production Contract
Major Products
Materiel Solution
Prototypes
Production Representative Articles
Low-Rate Initial Production Systems
Full-Rate Production Systems
Logistics/ Sustainment
MDD
A
B
Post-CDR A
C
FRP DR
Defense Acquisition System
(event-driven)
Technical
Financial Management
Cost Estimation Methods
Analogy
Parametric
Engineering
Actual Costs
Types of Funds
RDT&E – Advanced Technology Development
RDT&E – Advanced Component Development and Prototypes
RDT&E – System Development & Demonstration
Procurement
Operations and Maintenance
MILCON – Management & Support
Appropriated Funds To Support Contracts
Planning, Programming, Budgeting & Execution Process
(annual-calendar-driven)
Military Departments and Defense Agencies
Office of the Secretary of Defense and Joint Staff
White House
POM/Budget Formulation
Defense Planning & Programming Guidance
Integrated Program/Budget Review
Decisions
MBI
DoD Budget
President's Budget to Congress
DoD Testimony
DoD Appeals
Budget Committees
Authorization Committees
Appropriation Committees
Authorization/ Appropriation Acts Passed
Congressional Budget Process
Allocation
Apportionment

Building and Testing Virtual Assistants

January 27, 2020

Objectives

- Understand the design of software development paradigms on VA creation
- Apply modern paradigms to VA creation
- Develop team structure and roles
- Understand typical deliverables per sprint of a 4 sprint project

Overview

Software Development Lifecycle

- Waterfall
- Agile
- Waterfall vs. Agile

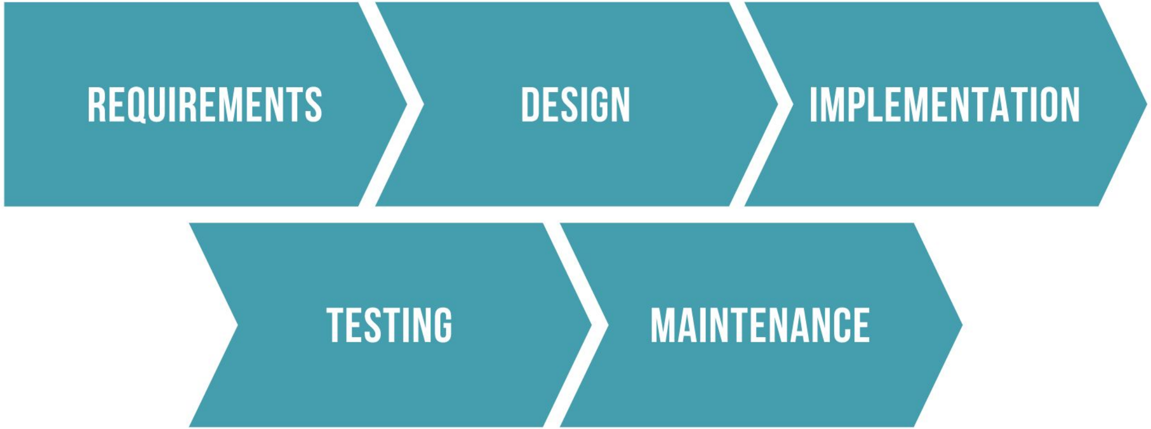
Test Driven Development

- Combining TDD and Agile
- Bootstrapping Regression Tests
- Accelerating Build Timeline

Software Development Lifecycle

Waterfall (Generic Methodology)

- Easy to plan
- Sequential tasking
- Clearly defined stage



REQUIREMENTS

DESIGN

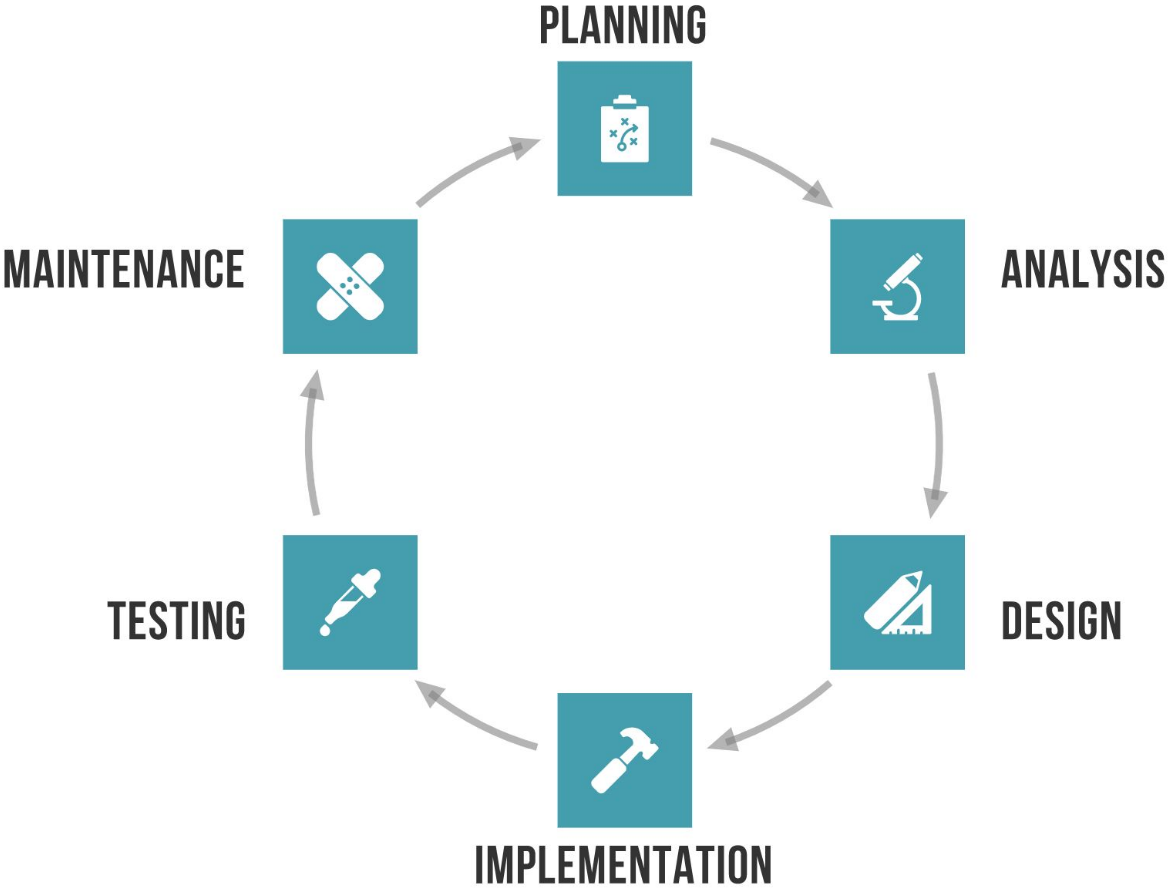
IMPLEMENTATION

TESTING

MAINTENANCE

Agile Methodology

- Improved quality
- More predictable delivery
- Increased flexibility



Agile vs. Waterfall

Agile

+ Easy to understand
+ Clear stages
- Resource bottlenecking
- Slow to correct errors

Waterfall

+ Enables continuous improvement
+ Improved stakeholder engagement
+ Full resource utilization
- More overhead than waterfall
- Less holistic view

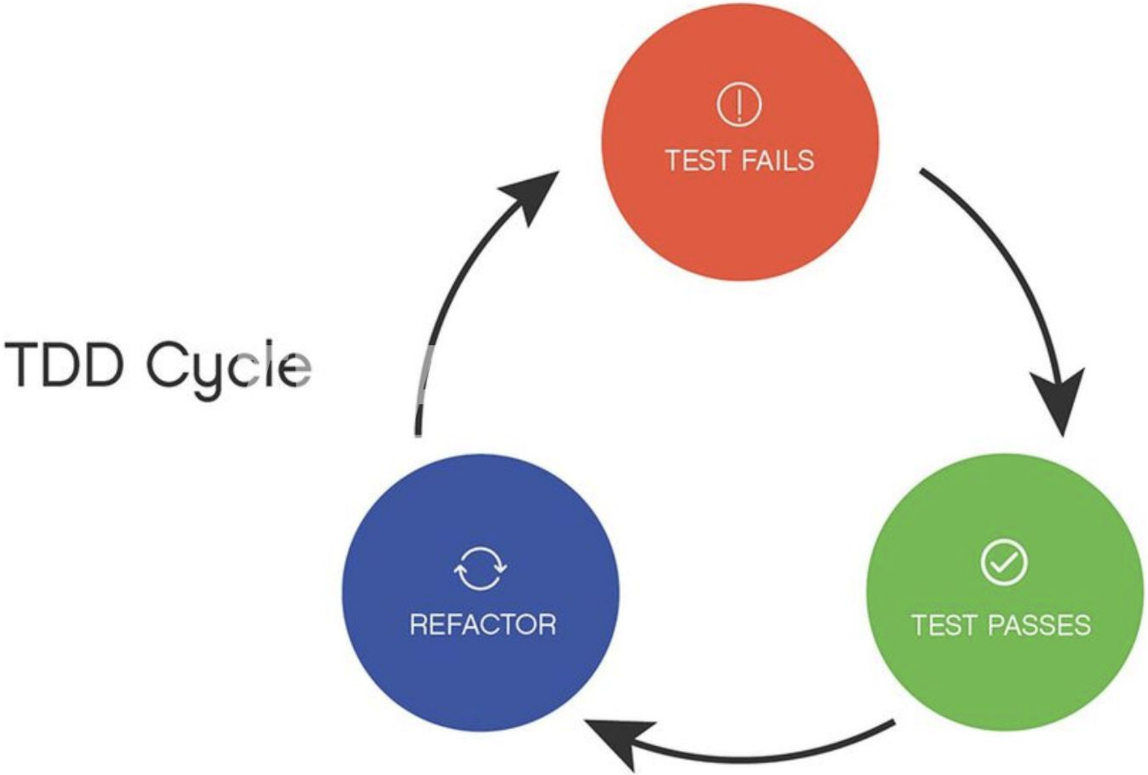
What Can We Do Better?

- Reduce time to test
- Improve inter-sprint scope perseverance
- More quickly bring all resources to bear (developers, testers, integrators, etc.)
- Validate regression conformance in cycle

Test Driven Development

TDD

- Guarantee test coverage
- Validate regressions
- Iterate on portions of the workflow



Combining Agile and TDD

Live each user story

- Actually try it
- Document edge cases
- These will be your tests!

No area left behind

- Each sprint should focus on all aspects of development (CLF, SVP, SM, BL)
- Progressively culminate more and more tests, making each sprint more refined

What is a Test?

What Can We Test?

Branching

- Classification
- Logic Driven
- Slot Fulfillment

Slot/Entity

- Extraction
- Disambiguation

Responses

- Channel customization
- Validation

Branching

Measures

- Containment Rate: *percent of conversations that are successfully completed within the VA*
- Precision: *correct classification:total test assertions*

How We Test It

- End-to-End tests of entire conversations
- Classification unit tests
- Query log review

How We Can Improve

- Define and revise scope
- Validate responses drive actions
- Review multi-turn conversations for alignment of prompt and response
- Classification insights (stop words, uniqueness)

Slots/Entity Extraction

Measures

- Precision: *extracted values:correctly extracted values*
- Recall *extracted values:present values*
- Context Retention: *retained:correctly presented*

How We Test It

- Test dialogs (SVP)
- End-to-End tests of entire conversations
- Query log review

How We Can Improve

- More slot data?
- Labelling insights
- Unlabelled data

Responses

Measures

- Containment rate: *percent of conversations that are successfully completed within the VA*
- Various qualitative measures: *severity, sentiment, etc.*

How We Test It

- User Acceptance Testing
- Sentiment Analysis

How We Can Improve

- Refactor responses
- Better define out of scope messaging
- Better edge case handling
- Ensuring prompts and outgoing transition alignment

Sprint Planning

Project Planning - Timelining Example

End of Sprint 1

- ➔ Skeleton of competencies on state graph with transitions
- ➔ First coupe waves of data collection completed
- ➔ Initial attempt at wireframes for web/mobile app
- ➔ Develop QA test suites: one for each competency, multiple demo flows etc.

End of Sprint 2

- ➔ Intents are being classified and slots extracted somewhat consistently (except for outliers)
- ➔ Continued collection & curation of data
- ➔ Wireframe to template code
- ➔ Basic responses created
- ➔ Framework of business logic and necessary API integrations has been created

Project Planning - Timelining Example

End of Sprint 3

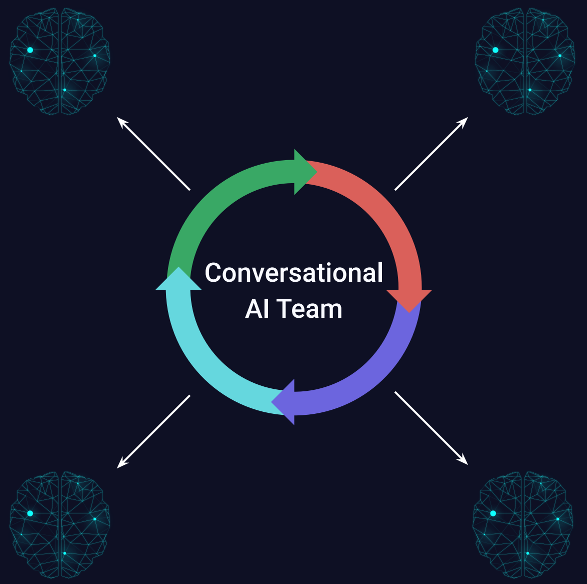
- ➔ Heavy focus on curation of data this sprint
- ➔ Classification & SVP should be near desired accuracy (>90%)
- ➔ Responses edited
- ➔ Business logic in progress
- ➔ Validated regressions

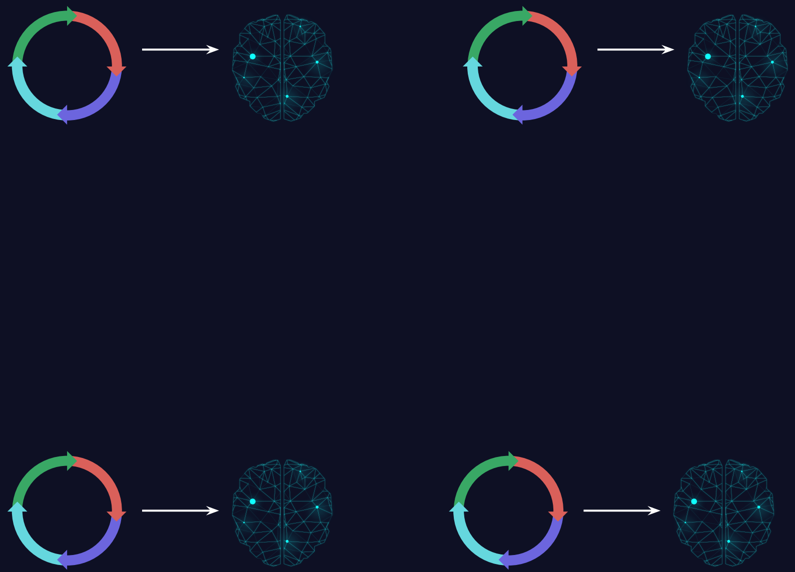
End of Sprint 4

- ➔ All test suites are passing for:
  - ◆ Classification
  - ◆ SVP
  - ◆ Business logic/responses
- ➔ Front end integration completed
- ➔ Team assesses finished product mid-sprint
- ➔ Demo to stakeholders

Team Composition

Team Composition





Centralized:
Single large team serving multiple LOBs
+ Centralized experience, efficient from a personnel perspective
- Large team overhead, slower delivery, synchronization needed

Decentralized:
Multiple small teams part of each LOB
+ Faster delivery, closer to the experience
- Fragmented experience, each team requires XO/CD/SW/CE

Roles and **Respective** Responsibilities

Product Owner

Business scope definition & iteration
Usage & containment analysis
Out-of-scope analysis
Channel growth analysis

Conversational AI Designer

Personality creation
Voice of the enterprise
UI/UX creation

Software Engineer (SW)

Deployment
Load & resilience testing
Channel integration
Business logic

Conversational AI Engineer (CE)

Data quality analysis
AI enhancements
Data collection & labelling
Quality assurance
Technical scoping

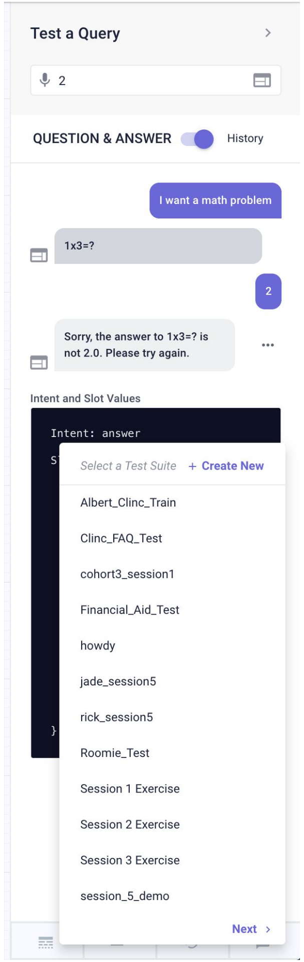
Project Manager

Clinc Tips

Save Your Conversations!

Testing will happen regularly during building.

- Collect conversations in the sidebar.
- Save conversations to test suites
- Run test suites at the end of sprint



Test a Query

2

QUESTION & ANSWER History

I want a math problem

1x3=?

2

Sorry, the answer to 1x3=? is not 2.0. Please try again.

Intent and Slot Values

```
Intent: answer
S
}
```

Select a Test Suite + Create New

Albert_Clinc_Train

Clinc_FAQ_Test

cohort3_session1

Financial_Aid_Test

howdy

jade_session5

rick_session5

Roomie_Test

Session 1 Exercise

Session 2 Exercise

Session 3 Exercise

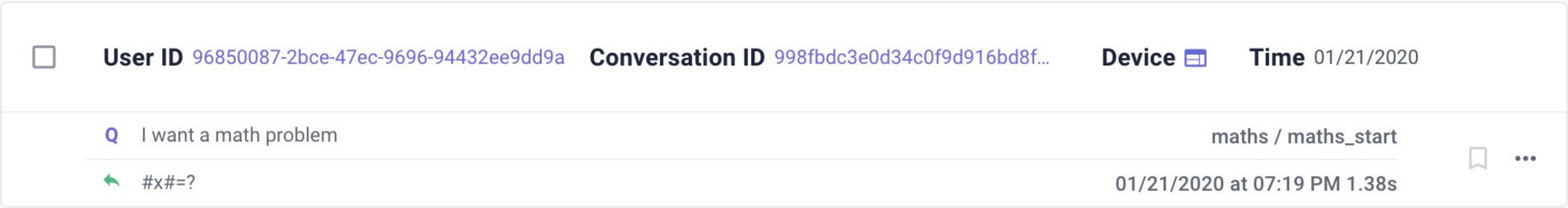
session_5_demo

Next ›

Leverage Real User Interactions

User interactions are tests NOT training!

- Adding failed CLF or SVP directly to training will leave blind spots
- Correct and add failed utterances to tests
- Curate data to pass curated tests



□ **User ID** 96850087-2bce-47ec-9696-94432ee9dd9a **Conversation ID** 998fbdc3e0d34c0f9d916bd8f... **Device** **Time** 01/21/2020

Q I want a math problem — maths / maths_start

#x#=? — 01/21/2020 at 07:19 PM 1.38s