Modern Testing (MT)

- All Testing episodes to listen
  - PARKING LOT - Place for notes and/ideas/ homes
    - testers ship it, sharpsorting people at local ship restaurant
    - in market are black bars is begging up to that point and it's been what you wanna to start
    - Modern testing principles not modern testers principles
    - automation is boring
    - automation is essential too
    - automation especially U-block holes of pagerdi
  - Episode 0
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-00-becoming-mtm/
  - Episode 1
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-01-production-is-the-tester/
  - Episode 2
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-02-the-mt-principles-and-practices/
  - Episode 3
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-03-the-mt-principles-and-practices-part-2/
  - Episode 4
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-04-the-mt-principles-and-practices-part-3/
  - Episode 5
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-05-testing-is-a-community-principle/
  - Episode 6
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-06-the-mt-principles-and-practices-part-4/
    - more episode 6
      - 6.1
      - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-06.1-customers-bugs-and-manglers/
  - Episode 7
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-07-the-mt-principles-and-practices-part-5/
  - Episode 8
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-08-testing-isn't-dead-testers-aren't-dead/
  - Episode 9
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-09-the-mt-principles-and-practices-part-6/
    - Episode 9.1
      - 9.1
      - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-09.1-the-corruption-of-the-modern-testing-principles/
      - general rates is customer-driv principles?
    - Episode 9.2
      - 9.2
      - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-09.2-digging-in-on-modern-testing-principles/
  - Modern testing in Context driven
    - 10
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10-a-modern-testing-meets-context-driven-testing/
  - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10-a-modern-testing-meets-context-driven-testing/
  - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10-special-delivery-quotation/
  - Transition guide
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10.5-the-quality-culture-transition-guide/
  - Lock
- principles
  - https://www.anyguyawaal.com/diff/testing/modern-testing-principles/
  - WHAT is IT ON WHO WHY THAT AS THEN?
    - MT is trending have to get "THERE"
      - Moving back from one center to one benefit
      - Being Force multiplier
    - Why Do We have MT Principles?
      - Software is much easier to produce
      - Thanks to advancements in Continuous Delivery, the customer has much more options to choose from.
      - Companies are dripping faster
      - It's much easier to manipulate with any company
      - For customers, the switching cost has gone way down
    - The principles are not just theory
      - They are based on Experience and observation
    - Why modern testing
      - To make contrast with Traditional
      - Traditional testing is not only a core but also delay
    - 1.0 World is changing and Traditional Testing is not catching up
      - Jeg time there is a change there is opportunity
  - Principle 1
    - Our priority is improving the business
    - What it means to improve the business?
      - MT or needs to be seen as positive influence
      - Testing Bugs
        - Adoption has to be product
      - Sales from Idea to deployments is rapid
      - Cut that step not with just enough quality to get developers from customers
      - You don't get value from engineering effort until it is in customer hands
        - Think not only how to get it out but what's own
      - Are one every your product better than your customer
        - what in case of Software Houses? They don't have product per se
      - There are bugs and bugs that customer sees about
      - The Bible
        - Short Term
          - Provide Value to customer Frequently
        - Long Term
          - Making sure Principle are Followed
      - Our goal is to accelerate team
        - that doesn't mean "we have to go fast"
        - Instead we are force multiplier
      - WHO
        - Adaptability
        - Reaction
        - no about
        - 2x
          - Brent warm on Lagunes and never finished his triangle
        - 1. Identify Value
        - 2. Map the Value Stream
        - 3. Create Flow
        - 4. Establish Pull
        - 5. Seek Perfection
        - Lean Principles
          - Deliver Sam by managing the flow
          - source: https://www.lean.org/WhatIsLean/Principles.cfm
      - WHY
        - Books to read
          - The Goal: A Business Graphic Novel
          - Phoenix Project
          - Lean Thinking
        - Don't forget MT mesure
          - By honoring the MT Pseudonear on identifying that role as productive?
            - this model were chosen because they are proven
          - It's important cause he is much of The 1002 is missing them
            - Way I name the Bottleneck Instead of resolving them
            - Example: stabilization period just before release
            - Or wait on "Audit quality of" instead of trying to see "quality in"
            - No I don't mean the test
            - Two Complaints date
              - "If you ever after test complaints you will find bugs and that will cause to no into the deadline"
      - HOW
        - This mantra is about complacency and predictability not about value it order to accelerate you need mechanism to identify what value is and what the core bottleneck
          - Mr dont want testing to be viewed as books rock
        - Quality is problem actual in humans being
        - Fast feedback loop is critical in course of understanding what is quality
        - By Mastering Value of value with failure in launch mechanism and Mastering Value of Value
        - "We are going to Sector continuously adapting and improving whose safety net presentation model?"
          - Thou shall SIX your safety net
            - Traditional testing is treated as safty net
        - In serving that stories fuel validated by the Fact the developers don't want to test
          - If we are going to move quality upstream, we have to remed shortcomings without the safty net
            - "Thou many people think the quality is limited to testing"
          - The tester can coach code cornerstones, but does own it
        - Code cornerstones
          - Continuous improvement is about not being afraid of failure and instead embracing them as a learning opportunity
          - Developers have to learn from their mistakes.
          - tech.smi testing a safety net test?
        - Safty net is a problem
          - Brent was harness lead in developers stepped using Cucumber and Selenium to Ruby and Capybara. Cause automatedt assert were Invasive, - its terrible cause sleepy architectures we't scale and is hard to maintain
          - Example
            - To remove safty net, you need also ensure mid trial and Mastering Value of consequences of mainage cannot be solve
            - Continuous improvement
              - getting better all the time
                - Every time, "****" every half year, "****" every two weeks
                - aka. TAE TAM
              - embraceable so sarty
              - Kaizen
                - looking for improvement every time everywhere
                - reducing the waste
                - Efficiency vs productivity
  - Principle 2
    - We assume we are blind, and use models like Value Stream, Kaizen, and Continuous Improvement to identify, prioritize and instrument bottlenecks from the outside.
  - Principle 3
    - We are a force for continuous improvement. We continuously improve our practices and our tooling. We learn from our mistakes and help others to learn following.
    - WHY
      - Vs a critical process change directly related to Quality
      - Testers are well suited to drive continuous improvement
        - Tester has to raise the system as a whole because of this they can easily spot place for improvement
        - Testers are specialists in asking questions
      - HOW
        - By measuring
          - Visibility
        - Unified engineering
        - Lots of small changes
        - Frequent retrospective (why 1.5 weeks)
          - Testers are well suited for driving retros
          - also useful model for long term retros
        - A small vertical slices of architecture
          - MVP
        - Master branch owned by the business they can release to customer this days
        - Quality culture and leadership
  - Principle 4
    - We test deeply where Risk Quality outcomes of our work, and we create, share, and continuously improve our tooling to develop a strong quality culture.
    - WHAT
      - Testers need to stop being passive
        - Quality
          - is not Bugs
          - Its not code cornerstone
          - Its not "mission to what you built
            - What is a higher quality app with no bugs but with like rarely or app with lot of bugs but 100x users?
          - is above eliminating friction (enhancing positive outcomes) Engineered
            - A
              - Low Confidence
              - The shared mindset that delivering high quality software is a team effort. Tooling, engineering and all test practices support this effort
            - B
              - Strong Confidence
              - It's how do we to build quality, actions, not reaction based, useful rules of users empathy within the organization using whose problems that users have
        - Quality culture
          - Getting people closer to understanding the problems that users are facing, how software developers are geared to solve it and how to stay current.
            - Getting people closer to understanding the problems that users are facing, how software developers are geared to solve it and how to stay current.
            - "Stay, customer are you happy?"
          - What it means Quality culture
            - Is a maturity model use with caution
              - Its also model. It doesn't has to work for you
            - Testing breadth
              - infancy example
                - functional testing
              - adulthood example
                - contextual testing
            - Quality over ownership shift
              - infancy example
                - Tester are doing all the testing.
              - adulthood example
                - Testers are coaches
                - No testers needed, never mant gate it
            - Bug this cycle
              - infancy example
                - bug backlog
              - adulthood example
                - zero bug policy
            - how do you use code cornerstones tools
            - Data analysis
              - infancy example
                - oblivious
              - adulthood example
                - data centric
            - Development approach
              - infancy example
                - ad-hoc
              - adulthood example
                - super small pushes
            - Learn and improvement
              - infancy example
                - not learning happens
                - retrospectives are for whining
              - adulthood example
                - every failure is opportunity to learn
                - and opportunity is value
            - Leadership support
              - infancy example
              - adulthood example
                - good quality practices are praised supported and rewarded
                - Bend leadership on the line
                - manifesto
      - ADKAR
        - Source
          - https://www.prosci.com/adkar
        - Awareness
        - Desire
        - Knowledge
        - Ability
        - Reinforcement
        - A AWARENESS OF THE NEED FOR CHANGE
        - D DESIRE TO SUPPORT THE CHANGE
        - K KNOWLEDGE ON HOW TO CHANGE
        - A ABILITY TO DEMONSTRATE SKILLS & BEHAVIORS
        - R REINFORCEMENT TO MAKE THE CHANGE STICK
      - WHY
        - The reason is we want feedback from our customers, more often
          - we want to understand and react to what they are doing.
          - we want to reduce risk in the ability to do that
        - To work in the way we want to function, to keep, retain and engage more users we need the quality culture
        - so improvearning
          - Fuse Risk minimalization
          - so learning to immediatly garnish
      - HOW
        - Unlike the Infinit team has a common understanding of the problem trying to solve it is possible
        - Small Batches
        - Knowledge sharing (software is knowledge works)
        - Try stuff + fail + learn
          - Experiment
            - Don't make sweeping changes at once
        - Reaping the harvest of events only when people are ready
          - Small changes, and dont give up
            - Don't try to change all at one
        - Brent example on stack TC.100 series
          - big heavy: move time on QA, that as dev, now DO, I seem to die as a manager. Half of my time were dev and half of my time were manager. Half of the things I did in year 0 bug backlog pulsated. Jumped up and down 100, and daily. bugs will not be fixed by the week profit.
          - After learning/unlelling period was over, the whole team remembered they would not ever go back to old ways.
  - Principle 5
    - We believe that the customer is the only valid measure of quality, so we gather and analyze the quality of their product.
    - WHAT
    - WHY
    - HOW
  - Principle 6
    - We use data extensively to deeply understand our customers' usage and their needs to guide innovation, action hypotheses, and business impact.
    - WHAT
    - WHY
    - HOW
  - Principle 7
    - We respond learning attitudes and inventive among our customers, our teams, our tools, our products, our processes, and our mind. We is model such learning operation.
    - WHAT
    - WHY
    - HOW
- Link to map
  - https://www.canva.com/ai/chat
- FAQ
  - Agile is more fundamental than MT, and addresses generic issues. MT principles are more focused and addressing testing issues.
  - These principles sound like Agile things. Why are they needed and separated from Agile?
  - How is agile/scrum
    - Reducing waste and increasing value
    - There is no MT specialist but MT activities are part of unified development. MT should not use tools PM but from every in every if needed.
    - Follower of MT mirrors pattern
  - Most of them sound more like managerial principles than testers.
  - https://tarasatz.wordpress.com/2010/10/mtm-manifesto-and-principles/#comments-1
    - unified development definition
- How it aplies to polish market?
- Historical information
  - General
    - Based on agile principles
      - 1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
      - 2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
      - 3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
      - 4. Business people and developers must work together daily throughout the project.
      - 5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
      - 6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
      - 7. Working software is the primary measure of progress.
      - 8. Agile processes promote sustainable development. 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—the art of maximizing the amount of work not done, is essential.
      - 11. The best architectures, requirements, and designs emerge from self-organizing teams.
      - 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
    - Who is customer for MT Principles?
      - Embed in this service
      - Tester as coach
      - Modern Tester
        - Who is the Champion of quality
        - Responsible for quality culture
    - Principle importance
      - Agile think all are equally important
      - Brent thinks principle 1 is most important
    - Episode 0? - 9?
    - Principlesnot first version
      - The customer is the only one complete to judge and evaluate the quality of our software.
      - The apply the theory of constraints to identify, prioritize and mitigate bottlenecks (from the system) and accelerate the teams.
      - We are not the safety net for software cornerstones and focus on improving the business, and not the tester.
      - The one responsible for the quality culture of our team, and are accountable for holding and creating the team in this regard.
      - We use tools to deeply understand customer usage and customer pain.
      - Embrace continuous improvement, and help the team adapt and optimize in order to solve customer pain.
      - The strive to reduce or eliminate the need for a separate testing phase, and instead focus on increasing testing abilities and initiatives across the team.
  - Legacy Principles
    - Legacy: current version doesn't have modern tester role yet
    - Software quality/code cornerstones
      - It is combination of Testing Discipline , agile and Kaizen as idea
      - Testing is not responsible for codecornerstones
      - Concerns to analyzes of breaks tests
        - Serving often to a breaks tests but safty is a mom cause of pain
      - Code cornerstones
        - Code is not the safety net for software cornerstones
        - We are not the safety net for software cornerstones
        - Code is not responsible for codecornerstones
      - As about failures
        - neither it doesn't need to be now, nor it doesn't have to be perfect. Jesterer.
- traditianal vs modern debate
  - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-0.1/

Legend
- Answer that is more funny / re...
- Outdoor
- Quote
- Needs more infancy questions
- Attention
- Materials already added to map

Modern Testing (MT)

- All Testing episodes to listen
  - PARKING LOT - Place for notes and/ideas/ homes
    - testers ship it, sharpsorting people at local ship restaurant
    - in market are black bars is begging up to that point and it's been what you wanna to start
    - Modern testing principles not modern testers principles
    - automation is boring
    - automation is essential too
    - automation especially U-block holes of pagerdi
  - Episode 0
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-00-becoming-mtm/
  - Episode 1
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-01-production-is-the-tester/
  - Episode 2
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-02-the-mt-principles-and-practices/
  - Episode 3
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-03-the-mt-principles-and-practices-part-2/
  - Episode 4
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-04-the-mt-principles-and-practices-part-3/
  - Episode 5
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-05-testing-is-a-community-principle/
  - Episode 6
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-06-the-mt-principles-and-practices-part-4/
    - more episode 6
      - 6.1
      - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-06.1-customers-bugs-and-manglers/
  - Episode 7
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-07-the-mt-principles-and-practices-part-5/
  - Episode 8
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-08-testing-isn't-dead-testers-aren't-dead/
  - Episode 9
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-09-the-mt-principles-and-practices-part-6/
    - Episode 9.1
      - 9.1
      - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-09.1-the-corruption-of-the-modern-testing-principles/
      - general rates is customer-driv principles?
    - Episode 9.2
      - 9.2
      - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-09.2-digging-in-on-modern-testing-principles/
  - Modern testing in Context driven
    - 10
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10-a-modern-testing-meets-context-driven-testing/
  - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10-a-modern-testing-meets-context-driven-testing/
  - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10-special-delivery-quotation/
  - Transition guide
    - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-10.5-the-quality-culture-transition-guide/
  - Lock
- principles
  - https://www.anyguyawaal.com/diff/testing/modern-testing-principles/
  - WHAT is IT ON WHO WHY THAT AS THEN?
    - MT is trending have to get "THERE"
      - Moving back from one center to one benefit
      - Being Force multiplier
    - Why Do We have MT Principles?
      - Software is much easier to produce
      - Thanks to advancements in Continuous Delivery, the customer has much more options to choose from.
      - Companies are dripping faster
      - It's much easier to manipulate with any company
      - For customers, the switching cost has gone way down
    - The principles are not just theory
      - They are based on Experience and observation
    - Why modern testing
      - To make contrast with Traditional
      - Traditional testing is not only a core but also delay
    - 1.0 World is changing and Traditional Testing is not catching up
      - Jeg time there is a change there is opportunity
  - Principle 1
    - Our priority is improving the business
    - What it means to improve the business?
      - MT or needs to be seen as positive influence
      - Testing Bugs
        - Adoption has to be product
      - Sales from Idea to deployments is rapid
      - Cut that step not with just enough quality to get developers from customers
      - You don't get value from engineering effort until it is in customer hands
        - Think not only how to get it out but what's own
      - Are one every your product better than your customer
        - what in case of Software Houses? They don't have product per se
      - There are bugs and bugs that customer sees about
      - The Bible
        - Short Term
          - Provide Value to customer Frequently
        - Long Term
          - Making sure Principle are Followed
      - Our goal is to accelerate team
        - that doesn't mean "we have to go fast"
        - Instead we are force multiplier
      - WHO
        - Adaptability
        - Reaction
        - no about
        - 2x
          - Brent warm on Lagunes and never finished his triangle
        - 1. Identify Value
        - 2. Map the Value Stream
        - 3. Create Flow
        - 4. Establish Pull
        - 5. Seek Perfection
        - Lean Principles
          - Deliver Sam by managing the flow
          - source: https://www.lean.org/WhatIsLean/Principles.cfm
      - WHY
        - Books to read
          - The Goal: A Business Graphic Novel
          - Phoenix Project
          - Lean Thinking
        - Don't forget MT mesure
          - By honoring the MT Pseudonear on identifying that role as productive?
            - this model were chosen because they are proven
          - It's important cause he is much of The 1002 is missing them
            - Way I name the Bottleneck Instead of resolving them
            - Example: stabilization period just before release
            - Or wait on "Audit quality of" instead of trying to see "quality in"
            - No I don't mean the test
            - Two Complaints date
              - "If you ever after test complaints you will find bugs and that will cause to no into the deadline"
      - HOW
        - This mantra is about complacency and predictability not about value it order to accelerate you need mechanism to identify what value is and what the core bottleneck
          - Mr dont want testing to be viewed as books rock
        - Quality is problem actual in humans being
        - Fast feedback loop is critical in course of understanding what is quality
        - By Mastering Value of value with failure in launch mechanism and Mastering Value of Value
        - "We are going to Sector continuously adapting and improving whose safety net presentation model?"
          - Thou shall SIX your safety net
            - Traditional testing is treated as safty net
        - In serving that stories fuel validated by the Fact the developers don't want to test
          - If we are going to move quality upstream, we have to remed shortcomings without the safty net
            - "Thou many people think the quality is limited to testing"
          - The tester can coach code cornerstones, but does own it
        - Code cornerstones
          - Continuous improvement is about not being afraid of failure and instead embracing them as a learning opportunity
          - Developers have to learn from their mistakes.
          - tech.smi testing a safety net test?
        - Safty net is a problem
          - Brent was harness lead in developers stepped using Cucumber and Selenium to Ruby and Capybara. Cause automatedt assert were Invasive, - its terrible cause sleepy architectures we't scale and is hard to maintain
          - Example
            - To remove safty net, you need also ensure mid trial and Mastering Value of consequences of mainage cannot be solve
            - Continuous improvement
              - getting better all the time
                - Every time, "****" every half year, "****" every two weeks
                - aka. TAE TAM
              - embraceable so sarty
              - Kaizen
                - looking for improvement every time everywhere
                - reducing the waste
                - Efficiency vs productivity
  - Principle 2
    - We assume we are blind, and use models like Value Stream, Kaizen, and Continuous Improvement to identify, prioritize and instrument bottlenecks from the outside.
  - Principle 3
    - We are a force for continuous improvement. We continuously improve our practices and our tooling. We learn from our mistakes and help others to learn following.
    - WHY
      - Vs a critical process change directly related to Quality
      - Testers are well suited to drive continuous improvement
        - Tester has to raise the system as a whole because of this they can easily spot place for improvement
        - Testers are specialists in asking questions
      - HOW
        - By measuring
          - Visibility
        - Unified engineering
        - Lots of small changes
        - Frequent retrospective (why 1.5 weeks)
          - Testers are well suited for driving retros
          - also useful model for long term retros
        - A small vertical slices of architecture
          - MVP
        - Master branch owned by the business they can release to customer this days
        - Quality culture and leadership
  - Principle 4
    - We test deeply where Risk Quality outcomes of our work, and we create, share, and continuously improve our tooling to develop a strong quality culture.
    - WHAT
      - Testers need to stop being passive
        - Quality
          - is not Bugs
          - Its not code cornerstone
          - Its not "mission to what you built
            - What is a higher quality app with no bugs but with like rarely or app with lot of bugs but 100x users?
          - is above eliminating friction (enhancing positive outcomes) Engineered
            - A
              - Low Confidence
              - The shared mindset that delivering high quality software is a team effort. Tooling, engineering and all test practices support this effort
            - B
              - Strong Confidence
              - It's how do we to build quality, actions, not reaction based, useful rules of users empathy within the organization using whose problems that users have
        - Quality culture
          - Getting people closer to understanding the problems that users are facing, how software developers are geared to solve it and how to stay current.
            - Getting people closer to understanding the problems that users are facing, how software developers are geared to solve it and how to stay current.
            - "Stay, customer are you happy?"
          - What it means Quality culture
            - Is a maturity model use with caution
              - Its also model. It doesn't has to work for you
            - Testing breadth
              - infancy example
                - functional testing
              - adulthood example
                - contextual testing
            - Quality over ownership shift
              - infancy example
                - Tester are doing all the testing.
              - adulthood example
                - Testers are coaches
                - No testers needed, never mant gate it
            - Bug this cycle
              - infancy example
                - bug backlog
              - adulthood example
                - zero bug policy
            - how do you use code cornerstones tools
            - Data analysis
              - infancy example
                - oblivious
              - adulthood example
                - data centric
            - Development approach
              - infancy example
                - ad-hoc
              - adulthood example
                - super small pushes
            - Learn and improvement
              - infancy example
                - not learning happens
                - retrospectives are for whining
              - adulthood example
                - every failure is opportunity to learn
                - and opportunity is value
            - Leadership support
              - infancy example
              - adulthood example
                - good quality practices are praised supported and rewarded
                - Bend leadership on the line
                - manifesto
      - ADKAR
        - Source
          - https://www.prosci.com/adkar
        - Awareness
        - Desire
        - Knowledge
        - Ability
        - Reinforcement
        - A AWARENESS OF THE NEED FOR CHANGE
        - D DESIRE TO SUPPORT THE CHANGE
        - K KNOWLEDGE ON HOW TO CHANGE
        - A ABILITY TO DEMONSTRATE SKILLS & BEHAVIORS
        - R REINFORCEMENT TO MAKE THE CHANGE STICK
      - WHY
        - The reason is we want feedback from our customers, more often
          - we want to understand and react to what they are doing.
          - we want to reduce risk in the ability to do that
        - To work in the way we want to function, to keep, retain and engage more users we need the quality culture
        - so improvearning
          - Fuse Risk minimalization
          - so learning to immediatly garnish
      - HOW
        - Unlike the Infinit team has a common understanding of the problem trying to solve it is possible
        - Small Batches
        - Knowledge sharing (software is knowledge works)
        - Try stuff + fail + learn
          - Experiment
            - Don't make sweeping changes at once
        - Reaping the harvest of events only when people are ready
          - Small changes, and dont give up
            - Don't try to change all at one
        - Brent example on stack TC.100 series
          - big heavy: move time on QA, that as dev, now DO, I seem to die as a manager. Half of my time were dev and half of my time were manager. Half of the things I did in year 0 bug backlog pulsated. Jumped up and down 100, and daily. bugs will not be fixed by the week profit.
          - After learning/unlelling period was over, the whole team remembered they would not ever go back to old ways.
  - Principle 5
    - We believe that the customer is the only valid measure of quality, so we gather and analyze the quality of their product.
    - WHAT
    - WHY
    - HOW
  - Principle 6
    - We use data extensively to deeply understand our customers' usage and their needs to guide innovation, action hypotheses, and business impact.
    - WHAT
    - WHY
    - HOW
  - Principle 7
    - We respond learning attitudes and inventive among our customers, our teams, our tools, our products, our processes, and our mind. We is model such learning operation.
    - WHAT
    - WHY
    - HOW
- Link to map
  - https://www.canva.com/ai/chat
- FAQ
  - Agile is more fundamental than MT, and addresses generic issues. MT principles are more focused and addressing testing issues.
  - These principles sound like Agile things. Why are they needed and separated from Agile?
  - How is agile/scrum
    - Reducing waste and increasing value
    - There is no MT specialist but MT activities are part of unified development. MT should not use tools PM but from every in every if needed.
    - Follower of MT mirrors pattern
  - Most of them sound more like managerial principles than testers.
  - https://tarasatz.wordpress.com/2010/10/mtm-manifesto-and-principles/#comments-1
    - unified development definition
- How it aplies to polish market?
- Historical information
  - General
    - Based on agile principles
      - 1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
      - 2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
      - 3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
      - 4. Business people and developers must work together daily throughout the project.
      - 5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
      - 6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
      - 7. Working software is the primary measure of progress.
      - 8. Agile processes promote sustainable development. 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—the art of maximizing the amount of work not done, is essential.
      - 11. The best architectures, requirements, and designs emerge from self-organizing teams.
      - 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
    - Who is customer for MT Principles?
      - Embed in this service
      - Tester as coach
      - Modern Tester
        - Who is the Champion of quality
        - Responsible for quality culture
    - Principle importance
      - Agile think all are equally important
      - Brent thinks principle 1 is most important
    - Episode 0? - 9?
    - Principlesnot first version
      - The customer is the only one complete to judge and evaluate the quality of our software.
      - The apply the theory of constraints to identify, prioritize and mitigate bottlenecks (from the system) and accelerate the teams.
      - We are not the safety net for software cornerstones and focus on improving the business, and not the tester.
      - The one responsible for the quality culture of our team, and are accountable for holding and creating the team in this regard.
      - We use tools to deeply understand customer usage and customer pain.
      - Embrace continuous improvement, and help the team adapt and optimize in order to solve customer pain.
      - The strive to reduce or eliminate the need for a separate testing phase, and instead focus on increasing testing abilities and initiatives across the team.
  - Legacy Principles
    - Legacy: current version doesn't have modern tester role yet
    - Software quality/code cornerstones
      - It is combination of Testing Discipline , agile and Kaizen as idea
      - Testing is not responsible for codecornerstones
      - Concerns to analyzes of breaks tests
        - Serving often to a breaks tests but safty is a mom cause of pain
      - Code cornerstones
        - Code is not the safety net for software cornerstones
        - We are not the safety net for software cornerstones
        - Code is not responsible for codecornerstones
      - As about failures
        - neither it doesn't need to be now, nor it doesn't have to be perfect. Jesterer.
- traditianal vs modern debate
  - https://www.anyguyawaal.com/diff/testing/ab-testing-episode-0.1/

Legend
- Answer that is more funny / re...
- Outdoor
- Quote
- Needs more infancy questions
- Attention
- Materials already added to map