Dark Green - Add or Improve
Lighter Green - Continue
Yellow - undecided (yet)
Pink - No or not now (3 mos)
Red - Stop or definitely no.

Add stickies!
eg, to comment, to qualify or explain

Add stickies for related issues or concerns. May address later.

We start with these??
- One Team - One Vision (one product goal)
- Don't get distracted
- We have a Vision and Mission that is important
- Well and commonly understood
- Team members are Team Players
- Team inspirable
- Team inspired
- Team is Full Time
- Team thinks they can win
- Have sufficient skill set
- Impediments judged ok
- Support judged ok
- Team members want to "win", together
- Game w Rules
- Are we set up for success? Yes.
- Complementary skills
- No SPOF
- Less Silo'd w Time
- We all can help each other
- We will ask for help more, sooner

Wisdom of Teams
- A Real Team
- Small in Number (<10)
- Comple-mentary skills
- Committed to common purpose
- Performance goals
- A common approach
- Mutually accountable

IDEO
- HOT Team
- Totally dedicated to end result
- Slightly ridiculous deadline
- Team irreverent and non-hierarchical
- Well-rounded & respectful
- Eclectic space
- Flexibility
- Group work
- Brain-storming
- Empowered to get whatever needed

New New Prod Dev Game
- Built-in instability
- Self-organizing project teams
- Over-lapping development phases
- "Multi-learning"
- Subtle control
- Transfer of Learning
- Autonomy
- Compared to waterfall
- Multi-level
- Self-control
- Drive to learn
- Self-transcendence
- Sashimi approach
- Multi-functional learning
- Control through peer pressure
- Drive to share learning (knowledge)
- Cross-fertilization
- Control by love
- Create "std practice"

Levels of Work
- Epic-->
- Feature-->
- SS User Story -->
- Task (sub-task)
- Why break down?
- Need "bite size"
- Faster feedback
- More transparency re progress
- Know when to ask for help
- To support a game, with WINS
- To enable faster learning
- Better plan, better date guess
- Goals in Breaking up Stories
- Achieve 80-20
- Less work, more BV
- Chunks that enable Team productiv in building
- Epic, Feature, User Story: have BV too
- Tasks are "only" work
- Mapping of Epics, Features, Stories into...
- Sprints and Releases
- Multiple Planning purposes
- "When do I get to use my Feature?"
- Find best way to build and deliver BV
- aka "Sprint Mapping"

At start
- Plenty of impediments, but no known showstoppers
- List our impediments (subset could double Velocity)
- More fun
- The best 6 months of my career ever
- All Scrum Rules in SG
- Or: Most Scrum Rules
- A bare framework
- Framework incomplete; we must add
- BUT: process still KISS

Empirical Process
- Transparency, Inspect and Adapt
- We are always going to be getting better
- We are willing to try small experiments to get better
- Minimum Viable Bureaucracy
- So, in some sense, you are doing more than Scrum

Scrum Values
- Commitment
- Focus
- Openness
- Respect
- Courage

Roles, Meetings, Artifacts

Roles
- More stickies later
- PO
- SM
- Devs
- Chickens
- Business stakeholders
- Team Size = 7
- Scrum team members 100% and in ONE role
- Each person set up to do role decently

Scrum Meetings / Events
- Sprint Planning
- Team commits to reasonable work
- Daily Scrum
- Sprint Review
- Sprint Retrospective
- Sprint = 2 weeks

Artifacts
- PB/PBL
- PB Items
- SBL
- Sprint Goal
- Scrum Board
- Sprint Burn Down
- Release Burn Down
- Working Product
- DOD
- DOD Stronger (eg, zero bugs)
- Top 20 Impediment List

Other things
- User stories
- Story points
- Epics, Features, SSUS
- Track Velocity each sprint
- Avg Velocity over 3 spr's
- Expect velocity to be going up
- DOR
- DOR process
- PB Refinement
- Short term meeting
- Long term meeting
- Expectation that plan will change
- Team WIP limit during Sprint

Team Goals
- More fun / happiness
- Less stress
- Fewer Hours
- Higher quality
- More Velocity
- Incr 100% in 6 mons.
- More BV
- How much more BV?
- "This is the best 6 mos of my career"
- "I never want to leave this Team."
- Cust's are happier
- Note: Business Goals on a "different sheet"

Team Norms
- Time for DS
- Daily Scrum at 9:15
- Start - End Hrs
- Core Hrs
- 10-noon & 2-4pm
- Vacation Days
- How Comm

Decision-making
- PO gathers input re PBL order
- PO decides order
- **Anyone** can talk: one decider - quickly
- Anyone can jawbone PO
- How should other decisions be made?
- Method(s) of deciding
- Min Num of stories per sprint
- Rec: Min. 8 stories per Spr.

Decisions
- How do we do code review
- When fix code review issues
- Which Tech stack
- TDD?
- PP?
- All for one, and one for all
- Accountable to each other
- Other parts of XP?
- Sprint duration
- 2 weeks
- Idea: consistent (cadence)
- Sprint commitment
- At least 8 PBIs per Sprint
- Size of stories (in sprint) = avg size +/- range
- Init: Def of Reliable (70-80%)?
- Each person willing to support any stuck story or task (almost immediately)
- Later: 50-60% reliable
- Sprint carryover - handling
- Not counting stories added if space at end of Sprint
- Business stakeholders avail. (20%?)
- Manage Chickens
- One chicken per person?
- Report back to Team?
- See later
- Metrics
- Velocity
- Velocity graph
- Carryover - # of stories
- Happiness
- BVPs
- Bugs escaping sprint
- Quality - how?
- We expect Velocity to increase (100%) by ?
- Code reviews
- Using Imped List
- All types of impediments
- Add items to Imped List continuously
- Team may work on Imped via Sprint Bklg (sometimes)
- We expect managers to help
- Order: Mainly ROI + depend + other
- Track fixed impediments
- Team will prioritize impediments
- Fixers:
  - SM
  - Team
  - Others
- Team skill gaps
- Managers will ID and propose fill
- Team will ID and propose fill
- When? (Start, S3, every 4 sprints)
- Coding stds
- Team members help each other
- Team members ask for help sooner
- Team members accept help
- We define product success (at release points)
- We measure success vs Drivers
- Work on TOP problem now

Developers
- Performance approach
- 50% Team success
- 50% individual contribution
- Transparency, Openness, Honesty
- Discussion of issues
- Fail and Learn
- Experiment
- Some level of failure is good!
- be honest about it
- Willing to do whatever it takes
- Make clear in DS (or earlier) if stuck
- Person can ask for more help, up to everyone
- Share and collaborate
- Pairing
- "Missouri" - 15 min sharing (1 per week)
- Tech PIT - tech issue, 1 hour max
- Timebox for work
- Be reasonable
- 4+ hrs / day
- Max 40 hrs per week
- 2 hour tasks
- Minimize distractions
- For Everyone
- Key SM role
- Pomodoro technique
- List common distractions
- Other?

QA/Coders ratio
- Tests automated (80%)
- QA Tests cover much of UAT
- "Test First"
- Mini sprint "each day"
- Stories done in 2 days elapsed time
- Good DOD?
- All bugs fixed

PO
- PO decides order
- PO decides MVP
- PO ans Q's in Sprint quickly
- PO respon for DOR process
- All Q's answered before Sprint starts
- PO shoots for 80-20 (BVP to SP)
- PO should inc BV per release 50% in 1st yr
- PO 100% dedicated to Team
- PO manages interrupt work

Other
- Team know 20 ways?
- ...To breakdown stories
- Use of Scrum Board
- Min WIP
- Attack any stuck item
- Update "hourly"
- Which other visual mgmt techniques?

Values, Principles, Ideas
- How practiced is each V,P,I?
- See Scrum Values
- Commitment
- Focus
- Courage
- Respect
- Openness
- Agile Manifesto
- Individuals and interaction over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan
- See Agile Principles
- #1 #2 #3 #4
- #5 #6 #7 #8 #9
- #10 #11 #12

You can't see this image because it comes from a private source.
Manifesto for Agile Software Development
We are uncovering better ways of developing software by doing it and helping others do it. These are our values and
Agilemanifesto

Key Ideas
- The bad news doesn't get better with age
- If you wait for perfection, you might wait too long
- Fail fast (learn faster)
- When you come to a fork in the road, take it
- People are remarkably good at doing what they want to do
- Shut up and drive
- The Krispy Kreme theory
- All see same elephant
- Be disciplined
- More fun / happiness
- Accountable to each other
- Have fun regularly
- How well doing these?
- Do you do them better?

Activity
- Team eval at start: can we win?

Events in more detail
- Sprint Planning
- Create tasks in SPM (not before)
- Stories & Tasks
- DOR, right?
- "Speak Now"
- Sp on all "Sprint" stories
- Daily Scrum
- Daily
- Max 15m
- Biggest impediment
- Everyone listens
- Purpose: Be more successful
- Say what helps the Team
- Emergent leadership
- Sprint (itself)
- Working product by end each sprint
- Working product ~ each day in sprint
- This enables faster feedback (eg, by PO)
- Sprint Review
- Review (of situation)
- Good "customer" feedback (review)
- Full feedback
- Everyone can give
- Honest negative feedback
- Demo of Working Product
- PO resolves dis-agmts immed
- Feedback on BV
- Positive & Negative
- Retro - More
- Positives
- Negatives
- SM report
- ID top (4) impediments
- Use A3 approach

Artifacts, more details
- Product Backlog
- Stories for 3+ Sprints are sprint-sized
- PB goes out "1 year" (later - >bigger)
- Goal: 80-20
- Use INVEST
- Transparent
- Top 20 Impediments List
- Public
- Helps
- Prioritized
- Track When Created
- Work "1 at a time"Impediment
- Small (fixable in 1 sprint)

Req's Related
- Clearer requirements
- verbal, written, pictures, etc
- DOR process
- Devs expected to reject stories
- Minions help
- Document only what's needed
- New process helps?
- What works for **this** Team

PO - more
- PO pivots when needed
- PO also maintains course when appropriate

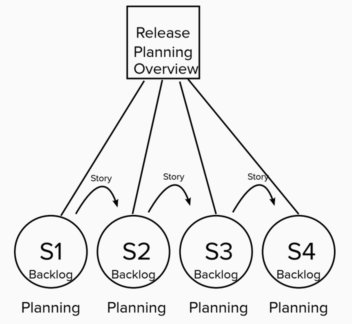
Ideas
- Team keeps an open mind re direction
- Stronger testing
- UT, FT, INT/ REG...
- Other types?
- Fix almost ALL identified bugs now
- Better continuous integration

More on PBL
- Prod Bklog
- First things first
- Prioritize mainly by ROI
- Include everything
- Prioritize everything

Ideas
- Single piece continuous flow
- Can you be closer to that?
- Planning always chasing change
- Adaptive planning
- Plan's always getting better - less inaccurate
- Plan's sometimes getting better for cust
- Pos & Neg change

Compared to Waterfall
- Doing "everything at once"
- Actively manage PB
- Build knowledge
- Build knowledge JIT
- Build our cust understanding
- Increase self-organization
- Swarming
- Other adaptations intra-sprint
- Incr Rate of innovation and creativity
- Build-in more Inno and Crea
- Define specific ways it will happen
- Discuss different levels (eg, in code, module, system design, implementation, bus solution, etc)
- Give recognition to people
- Rate current recognition "level"
- ID more ways to recognize
- Give more recognition
- Rate morale of team
- Include inspiration & motivation
- Include pos & neg
- ID how to improve morale
- Experiment
- ID any team member weaknesses
- ID many ways to improve
- ID ways to help them
- If a team member can't cut it, help that person transition

More on Imped List
- Top 20 Impediment List
- Share and collaborate across teams
- Build Collaboration
- With Bus side
- How?
- How much collab per week?
- More collaboration (time)
- Better collaboration (quality per min)
- How collab better
- How much knowledge sharing?
- Why?
- Better requirements
- Less wasted energy
- Faster delivery
- Result: More BV
- Result: Higher morale
- More "buzz" in team
- We will build it (how?)
- More rhythm, harmony
- STOPPED



Planning (con't)
- Planning over plans
- Time is important
- Smarter not harder
- More accurate plans (as we progress)
- Manage pressure / stress
- Key Activities
- Bring Scrum Team & BSHs
- One day for 6 mos
- Vision
- Product Backlog
- Roles (5-7)
- User Story Workshop (50)
- Business Value
- Drivers (3-5)
- Priority Poker
- Effort
- DOD review
- Planning Poker
- ROI = BVP / SP
- Other factors
- Risks, Depend, Learning, MMFS/MVP, other
- Identify expected Velocity
- Lay out sprints with Velocity
- ID number of sprints for a release
- Add contingency
- Add Landing Strip
- Calc avg cost per sprint
- Calc budget for 1st release
- Revising the plan every sprint
- ST Meeting
- Vote re details for each story
- Assuring DOR good before SPM
- LT Meeting
- Many activities to revise and refine the LT plan
- Anything in ARP can change, eg, the Vision
- Mainly: new stories, broken up stories, new BVPs, new SPs.
- Then consequences
- Try to make the plan better (more accurate)
- Try to make results better for customer

PB Refinement
- A. Moving toward Ready stories in Spr X
- B. Revising the plan; to learn and to make LT decisions
- B. And to get a better date / scope / budget trade-off
- Who does what?
- How do we add the details?
- Who leads breaking down and adding details?
- How do we do the break down?
- Who leads plan revision?

MISC
- Stop tracking worked hours to assigned story
- 2 hour tasks
- Do estimate hours remaining on tasks
- In DS you must say what you did and what you will do
- How revise SPs on stories
- Coders & Testers vote on tot effort
- Coders and Testers playing same game
- Team makes sure work is balanced across individuals
- We all are in it together
- Team compares SP to task estimates in SPM
- Happiness metric
- See: Jeff Sutherland's blogs - scruminc.com

Chickens
- Manage Chickens
- Who can a Chicken be?
- What help does the team need?
- Identify Chickens
- Identify work for Chickens
- Assign team member to Chicken
- Report back
- Escalate if needed
- Take other action
- How integrate better with UX group

PO - More
- PO Role
- Inspires
- Leads plan revisions
- Leads DOR process
- Dup: PO respon for DOR Process

SM - More
- SM Role
- SM 100% dedicated to 1 Team?
- Get impediments removed
- Min of continuous improvement
- Min of Truth and Transparency
- Min of self-org
- Min of self-managing
- Min of Collaboration
- Min Distractions
- Facilitate

Chickens - More
- Approach to working with Product grp
- Approach to working with other bus areas
- Improved communication (Team to business)

PBL - More
- Backlog Health
- How many sprints of "ready" stories should we have?
- How many "levels" of Ready should we think about?
- How much should our backlog "emerge"? Why?
- What would a "perfect" Prod Backlog look like?
- Size (in months)
- Number of stories
- Number of stories that are NOT moving up? (fewer)
- Good story card
- >4 months, mostly Epics
- 2-4 mos mostly Features
- <2 months, S.S. user stories
- Good sprint sized stories fill sprint.
- Add Acceptance criteria - when?
- Add details when?
- How do we give detailed knowledge to Team?
- Convo, Pictures, Lists, etc.
- How describe a better grooming process from Epic down?

PO - More
- Details: PO responsibility
- Minions help PO
- Who are "minions"
- How do my BSHs fit in?

Change
- Good & Bad change
- Recognize Chg (quickly)
- Max Good Mitig Bad
- Change can affect...
- Our Way of Working (Productivity + or -)
- Our Product (eg, re-order work, add work, chg direction)
- Other?
- What is our current pace of change? Hi or Lo
- Are we getting "blindsided"?
- Is Prod Direction being changed (good, ok, bad)?
- Make it make sense!

Interrupt Work
- someone proposes extra work in THIS sprint
- No!
- Later
- Even Exchange
- Interrupt Buffer
- Tell me Earlier!
- Remind: Managed by PO

Distractions & Changes in Direction
- aka: new priorities
- PO must sync with key stakeholders
- Vice versa
- PO seen as leading team
- Outsiders can announce new priorities once per sprint?
- New priorities are not for next Sprint! (norm)
- Often seen as distractions
- Often import. not easily seen
- Team needs to know, may change current work
- Avoid pivots 'til Curr Release out
- Discuss affect on Team morale
- Cf. Interrupt work
- Outcomes
- New stories
- Re-ordered stories
- Deprecated stories
- Increase or reduce BV
- Give up on part-built Release
- Faster or slower Delivery
- Response to competit pressures
- Team **needs** an explanation

The Top 10 Things

- Example of top 10 imporvements
- To be done 3rd day
- 1. Identify many things that might make list
- 2. Pick and order Top 10 "things" (may be "rows")
- #1
- Scrum roles
- Dedicated SM and PO for each team
- Have fun regularly
- Embrace change
- Rate morale of team
- Improve morale
- DOR
- DOD
- Define DOD
- DOD Stronger (eg, zero bugs)
- Working product by end each sprint
- Doing "everything at once" (i.e. Testing & UX included!)
- Transparency of "Other" Requirements (i.e., Tech Debt)
- Resolve infrastructure impediment for CI/CD
- Goal: 80-20
- Project Mission / Vision / Values
- Timebox for work
- Prioritize our stupidity
- Integrate better w/ UX group
- Document only what's needed
- Product Backlog > 1 yr
- Sprint carryover - handling
- Deliver more frequently
- Better CI / CD
- Tests automated (80%)
- UT & QA Tests
- Stronger testing
- Security Testing
- Accessibility Testing
- e2e testing
- One day for 6 mons
- Agile Rel Planning
- BVPs / SPs & ROI
- Track Velocity each sprint
- Stop tracking worked hours to assigned story
- Impediment List
- Engage better with DevOps
- Links to CI / CD