Agile Engineering Practices
planning vs doing

Go for the one that’ll beat the one you last did
planning is stage one

most agile methodologies ignore engineering

why does scrum hate developers?

developers gone wild?!?

agility ➔ discipline
feedback loops
automation
metrics
time & space
communication

non-
intuitivity
demonstration
which falls faster?
agile estimation [15 mins]

design practices [15 mins]

TDD & velocity [10 mins]

continuous delivery [10 mins]

DVCS magic [10 mins]

feature toggles [10 mins]

pair programming mechanics [15 mins]

why pair programming works [10 mins]

Version control
continuous delivery
DVCS magic
TDD & velocity
Try playing the throw that would have lost to your opponents last throw.

agile estimation
iteration 0
(inception)

architecture  QoS  testing

what does it do?  when?
estimation

for each story {
    BA/stakeholder describes what it does
    developers gauge complexity
    assign complexity points
}
complexity vs. time

rookie

\[ \text{time} = \text{coding with no interruption} \\
16 \text{ hours a day, subsisting on} \\
cold \text{ pizza \& mountain dew} \]

experienced developer

\[ \text{time} = 8 \text{ hrs} - (\#_\text{meetings} + \text{support\_calls} + \text{email} + \text{fixing\_printers}) \]
complexity

how complex is this story compared to other stories?
complexity

less ad-hoc variable values

more consistent across projects

gets better over time

builds trust
project manager assigns load factor to convert complexity to time
business chooses story order
estimation & metrics

quality of data → quality of metrics

course grained estimate by developers is a good starting place

project manager continuously gauges the quality of estimates using actual data

if the load factor is wrong → change it

if the estimates are poor → re-estimate
metrics enablers
business derived completion criteria

feedback loops

communication
Sample Story Card Request Manager to source widgets

VERSIONING & APPROVAL

<table>
<thead>
<tr>
<th>Version</th>
<th>Author/Modifier</th>
<th>Date</th>
<th>Changes</th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>BA Bob</td>
<td>3/1/2006</td>
<td>Updated after meeting with users</td>
<td>Approved</td>
</tr>
</tbody>
</table>

STORY

Provide sales the ability to mark as ready for review and send an email to Ops.

BUSINESS CONTEXT

When a sales manager is done with his/her work on an inquiry in terms of searching for and/or selecting widgets to it, operations needs to be informed to complete sourcing information on the order.

DEVELOPMENT RELEASE / ITERATION: RELEASE 1 - ITERATION 3

KEY PROCESS AREA: Order Inquiry

ANALYST: BA Bob

LIMITATIONS

<table>
<thead>
<tr>
<th>Story ID</th>
<th>Business Segment</th>
<th>Story Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>OI-20</td>
<td>Order Inquiry</td>
<td>Ask customer &amp; deliver information to order</td>
</tr>
<tr>
<td>OI-47</td>
<td>Order Inquiry</td>
<td>Define user roles for Order Inquiry and Order Management</td>
</tr>
</tbody>
</table>

EXISTING CONDITIONS:

Creating an inquiry and including customer information on the inquiry is possible in The Widget Project (OI-36, OI-3). Sending an inquiry is possible (OI-2) as well as selection of widgets on to an inquiry (OI-5a, OI-5b). Story OI-11a has been played which is the first story to setup an email alert.

STORY NARRATIVE (including examples):

The story begins from the point where a user has created a qualified inquiry where widgets may or may not have been selected.

1. Sales Manager is able to mark the inquiry as ready for review by operations.

On the Inquiry screen below the search results and widget selections, display this label 'Inquiry is ready for review by Manager and a submit button.

Once the submit button has been pressed, disable the button.

2. Associate Managers to Sales Managers

Author: BA Bob

Page 1 of 3

Last Updated: June 30, 2007
Sample Story Card Request Manager to source widgets

<table>
<thead>
<tr>
<th>Sales Manager</th>
<th>Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frank Dodati</td>
<td>Debbie Bone</td>
</tr>
<tr>
<td>John Martin</td>
<td>Debbie Bone</td>
</tr>
<tr>
<td>Mark Poeppe</td>
<td>Ellen Rioner</td>
</tr>
<tr>
<td>Paul Curry</td>
<td>Ellen Rioner</td>
</tr>
<tr>
<td>George Sullivan</td>
<td>Rick Leslie</td>
</tr>
<tr>
<td>Mike Keating</td>
<td>Rick Leslie</td>
</tr>
<tr>
<td>Tim Ewing</td>
<td>Maria Dabanovacki</td>
</tr>
<tr>
<td>Craig Newlin</td>
<td>Maria Dabanovacki</td>
</tr>
<tr>
<td>John Gynn</td>
<td>Margaret Ebert</td>
</tr>
<tr>
<td>Jeff Rasmussen</td>
<td>Keil Wisa</td>
</tr>
<tr>
<td>Bill Lyngard</td>
<td>Keil Wisa</td>
</tr>
<tr>
<td>Mike Casabucc</td>
<td>Laura Felix</td>
</tr>
<tr>
<td>Sonia Baumer</td>
<td>Alejandro Mendez</td>
</tr>
</tbody>
</table>

Store the names as First Name and Last Name.

3. Send alert to the associated Manager when inquiry is marked ready for review
   The email alert should follow the format described in DI-11a and contain the following information:
   Subject – Inquiry <inquiry name> is ready for review

   <salutation> < manager name >.

   Inquiry <inquiry name > for customer <customer name > created by <sales manager name > is ready for your review to complete sourcing and delivery information.

4. Once an inquiry is marked ready for review, allow only the Manager to change Inquiry criteria or widget selections if made.
   Maintain a "Ops Review Request Indicator" which will indicate that the inquiry has been sent to Ops for review. Also maintain the date on which the request is made. On the Inquiry screen, display "Ops Requested on <date >".

5. Add Ops owner to Search for Inquiry criteria

Validations (Include expectation for notifying user of invalid input)
None.

Audit Trail
All changes should be logged.

Security
Create a permission for the ready for review submission button.
Sample Story Card Request Manager to source widgets

Status: Approved

IMPACT TO OTHER SYSTEMS - INTEGRATION
None.

PERFORMANCE CONSIDERATIONS
None.

SCREEN MOCKUP (AS NEEDED)
Not required.

USER DOCUMENTATION/ONLINE HELP REQUIRED
Not at this time.

TESTS REQUIRED (INCLUDE ALL "HAPPY" ROUTES)
1. On qualified inquiries, the user is presented with a label 'Inquiry is ready for review by Manager' and a submit button.
2. Once depressed, the submit button is disabled.
3. On submission, an email in the desired format is sent to the Manager associated to the Sales Manager.
4. Check that only the Manager is able to change search criteria or widget selections once marked ready for review. Check that the date request was made displays.
5. Check that the Ops owner list shows up as a criteria in Search for Inquiry

ADDITIONAL TESTS
[Include additional testing covering other areas of the system that may indirectly be impacted by the changes in the above narrative] QA will update this area as they create the tests if necessary

REGRESSION TEST REQUIREMENTS
This test may be made available in the regression suite.
metrics binary completion

0 → 1.
project-level metrics
Sparky’s spreadsheet
## Use for the Iteration Kick Off Meeting

### Purpose:
Plan out how many folks you have vs. how much work you have, per iteration.

Developer team size and vacation days calculates from your budget sheet!

Note: This table determines your development capacity!

Changes to your expected velocity will change how much work you can expect to finish.

### Planning Iteration Work Load

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Development Start</th>
<th>Expected Load Factor</th>
<th>Team Sz. (Developers)</th>
<th>Developer Vacation Hours</th>
<th>Ideal Hours</th>
<th>ITKO estimates for the cards assigned to this iteration</th>
<th>Hangover Estimate (Unhide columns to include other Dev time constraints)</th>
<th>Over/Under</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tuesday, July 05, 2005</td>
<td>0.30</td>
<td>3.0</td>
<td>0.0</td>
<td>12.0</td>
<td>128.0</td>
<td>126</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>Tuesday, July 19, 2005</td>
<td>0.40</td>
<td>4.0</td>
<td>0.0</td>
<td>128.0</td>
<td>128.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>Tuesday, August 02, 2005</td>
<td>0.50</td>
<td>5.0</td>
<td>0.0</td>
<td>219.0</td>
<td>192.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>Tuesday, August 16, 2005</td>
<td>0.50</td>
<td>5.0</td>
<td>0.0</td>
<td>219.0</td>
<td>192.0</td>
<td>284.0</td>
<td>24.0</td>
</tr>
<tr>
<td>5</td>
<td>Tuesday, August 30, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>Tuesday, September 13, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>288.0</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>Tuesday, September 27, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>Tuesday, October 11, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>Tuesday, October 25, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>300.0</td>
<td>28.0</td>
</tr>
<tr>
<td>10</td>
<td>Tuesday, November 08, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>440.0</td>
<td>160.0</td>
</tr>
<tr>
<td>11</td>
<td>Tuesday, November 22, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>12</td>
<td>Tuesday, December 06, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>13</td>
<td>Tuesday, December 20, 2005</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>14</td>
<td>Tuesday, January 03, 2006</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>15</td>
<td>Tuesday, January 17, 2006</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>16</td>
<td>Tuesday, January 31, 2006</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>17</td>
<td>Tuesday, February 14, 2006</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>Tuesday, February 23, 2006</td>
<td>0.50</td>
<td>7.0</td>
<td>0.0</td>
<td>293.0</td>
<td>280.0</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**Iteration that just ended:** 4

**Capacity Remaining (Ideal Hours):** 1672

**Remaining Work in hours days, cards not Dev Complete:**

- A: 256
- A+B: 1388
- A+B+C: 1828

**Priority:**

- A
- A+B
- A+B+C

**Do you have too much work left, for the days you have remaining?** (Graph represented on the bottom of this page)
the customer is always available...

...or a worthwhile substitute
customer proxy

80% business, 20% technical

“feed” the development process

business analysts

(local) subject matter experts

instant answer source
requirements gathering is a lossy compression algorithm
design practices

Rock is for Rookies: males have a tendency to lead with Rock on their opening throw.
boring

anticipatory design

fear

why is simplicity hard?

cleverness mixed with

irrational attachment
choose a system metaphor

domain driven design’s ubiquitous language
class-responsibility-collaboration cards

alternative to UML

anything but UML!

use crc cards for design sessions

captures just what you need

deprecated by technology
design tools
what about...
documentation

useful

succinct

low ritual

DRY

tests!
create spike solutions to reduce risk

not prototypes!
no functionality added early

yagni
don’t build frameworks

extract them
case study
evolution of asynchronous messaging
progress bars & async upload

backgrounDrb
http://backgroundrb.rubyforge.org/
3 kinds
(Starling)

switch to a real messaging queue
don’t know what we don’t know

“buy the fanciest one we can” (just in case)
technical debt

when you add it

when you start using it

project time
don’t know what we don’t know

“buy the fanciest one we can” (just in case)

pay $$$ for technical debt...

...that you may never justify

project time
trying to predict the future leads to over-engineering
Scissors on First: play scissors as your opening move against a more experienced player.
code the unit test first

- Start
- Write a failing test
- Write code to make it pass
- Refactor
- Can't think of any more tests
- Stop

Red: green: refactor
test driven design

more about design than testing

design will emerge from tests

better abstractions

less accidental complexity

atomic understanding of intent
perfect number

case study

\[ \sum \text{of the factors} = \text{number} \]

(not including the number)
test-after, 1st pass

```java
public class PerfectNumberFinder1 {
    public static boolean isPerfect(int number) {
        // get factors
        List<Integer> factors = new ArrayList<Integer>();
        factors.add(1);
        factors.add(number);
        for (int i = 2; i < number; i++)
            if (number % i == 0)
                factors.add(i);

        // sum factors
        int sum = 0;
        for (int n : factors)
            sum += n;

        // decide if it's perfect
        return sum - number == number;
    }
}
```
public class PerfectNumberFinder2 {
    public static boolean isPerfect(int number) {
        // get factors
        List<Integer> factors = new ArrayList<Integer>();
        factors.add(1);
        factors.add(number);
        for (int i = 2; i <= sqrt(number); i++)
            if (number % i == 0) {
                factors.add(i);
                factors.add(number / i);
            }

        // sum factors
        int sum = 0;
        for (int n : factors)
            sum += n;

        // decide if it's perfect
        return sum - number == number;
    }
}
public class PerfectNumberFinder2 {
    public static boolean isPerfect(int number) {
        // get factors
        List<Integer> factors = new ArrayList<Integer>();
        factors.add(1);
        factors.add(number);
        for (int i = 2; i <= Math.sqrt(number); i++)
            if (number % i == 0) {
                factors.add(i);
                // guard against whole-number square roots
                if (number / i != i)
                    factors.add(number / i);
            }

        // sum factors
        int sum = 0;
        for (int n : factors)
            sum += n;

        // decide if it's perfect
        return sum - number == number;
    }
}
public class Classifier6 {
    private Set<Integer> _factors;
    private int _number;

    public Classifier6(int number) {
        if (number < 1)
            throw new InvalidNumberException(
                    "Can't classify negative numbers");
        _number = number;
        _factors = new HashSet<Integer>();
        _factors.add(1);
        _factors.add(_number);
    }

    private boolean isFactor(int factor) {
        return _number % factor == 0;
    }

    public Set<Integer> getFactors() {
        return _factors;
    }

    private void calculateFactors() {
        for (int i = 2; i < Math.sqrt(_number) + 1; i++)
            if (isFactor(i))
                addFactor(i);
    }

    private void addFactor(int factor) {
        _factors.add(factor);
        _factors.add(_number / factor);
    }

    private int sumOfFactors() {
        calculateFactors();
        int sum = 0;
        for (int i : _factors)
            sum += i;
        return sum;
    }

    public boolean isPerfect() {
        return sumOfFactors() - _number == _number;
    }
}
java.lang.AssertionError:
Expected: is <[1, 2, 3, 6]>
got: <[1, 6, 2, 3]>

at org.junit.Assert.assertNull(Assert.java:502)
at org.junit.Assert.assertNull(Assert.java:492)
at com.nealford.conf.tdd.perfectnumbers.Classifier3Test
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethod
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethod
at org.junit.internal.runners.TestMethod.invoke(TestMethod
at org.junit.internal.runners.MethodRoadie.runTestMethod(MethodRoadie
at org.junit.internal.runners.MethodRoadie$2.run(MethodRoadie$2.run)
```java
for (int i = 2; i <= Math.sqrt(number); i++)
    if (number % i == 0) {
        factors.add(i);
        // account for whole-number square roots
        if (number / i != i)
            factors.add(number / i);
    }
```

```java
private void calculateFactors() {
    for (int i = 2; i < Math.sqrt(_number) + 1; i++)
        if (isFactor(i))
            addFactor(i);
}
```

```java
private void addFactor(int factor) {
    _factors.add(factor);
    _factors.add(_number / factor);
}
```
case studies

Dr. Laurie Williams
Associate Professor
North Carolina State University
Department of Computer Science

http://collaboration.csc.ncsu.edu/laurie/publications.html
source: http://agile-carolinas.pbworks.com/f/WilliamsTDD.ppt
source: http://agile-carolinas.pbworks.com/f/WilliamsTDD.ppt
new “anti-aging” formula

source: http://agile-carolinas.pbworks.com/f/WilliamsTDD.ppt
writing more code allows you to go faster
pair programming mechanics

Paper is the least obvious of opening moves.
2 monitors

1 computer

2 mice

2 keyboards
pairing stations

not someone’s computer

all the tools for development...

...and nothing else

pairing station ≠ your laptop*

mirrored...
mirrored workstations

radmind

http://rsug.itd.umich.edu/software/radmind/
driver
navigator
logistics

driver types & narrates

navigator thinks & interjects

design discussions in situ

no discussion > 10 mins w/o code

swap roles frequently
pair rotation

twice a day ⇔ every other day

tech lead picks effective pairs

reduces truck number metric

spreads knowledge across team
1 person must stay with story

you can only stay once/rotation

context update for the new pair

swap

today’s new pair is tomorrow’s context keeper

promiscuous knowledge
not!
what it’s not!

2 people huddled over 1 computer

mentoring
learning  flow  mentoring

challenge  skill

overwhelmed  bored  coaching

learning
what it’s not!

2 people huddled over 1 computer

mentoring

keyboard domination
ping-pong pairing
what it’s not!

2 people huddled over 1 computer mentoring

keyboard domination

pair marriages

> 10 mins of debate with no code

less productive
pair programming studies

after adjusting, pairs produced code 15% more slowly than individuals...
pair programming studies

...with 15% fewer defects
Williams et al

pairs 15% slower

15% fewer bugs

“error free” code 70–85%

50% decrease in errors (30%–15%)

testing & debugging many times more $\$\$

http://www.economist.com/displayStory.cfm?Story_ID=779429
more studies

Lui 2006  

rigorous scientific experiment

novice–novice vs. novice solos

vs.

expert–expert vs. expert solo

novice Δ “significantly higher”

Lui, Chan, & Nosek: pairs outperform for design tasks

http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4378344
benefits
promiscuous knowledge

what I know
what u know
what we know
fungibility

domain knowledge

architectural understanding

design implications

keyboard shortcuts

effective tools
playing w/ better players
why pair programming works
left brain


right brain
left brain:

spoken language and writing
counting
rational thought and logic
analysis, recognition of details
governing and lawmaking
science
awareness of time
linear thought, "step by step"
left brain

right brain
right brain

body language
ability to visualize, daydreaming
intuition
synthesis, ability to synopsize
creativity, imagination
art, music, dance, color, rhythm
spacial awareness
holistic and non-linear thought
FLOW
THE PSYCHOLOGY OF OPTIMAL EXPERIENCE

MIHÁLY CSIKSZENTMIZSLEYI

“Flow couldn’t come at a better time for us. An inspiring, worthwhile read.”
—Chicago Sun-Times
time disappears

"in the zone"

tunnel vision

total concentration

insanely productive
Puma Productivity Pants™
cubicles make you dumber!
coding
!=
dull
managers

makers
overtime is bad
When playing with someone who is not experienced at the RPS, look out for double runs or, in other words, the same throw twice.

Thanks to my ThoughtWorks colleague Cosmin Stejerean for this topic.
feature branch
copy/paste
reuse !!

merge
ambush!
merge requires tests

[ textual

semantic ]
If it hurts...

... do it more often

bring the pain forward

feedback loops
Continual Integration
feature toggle

add configuration to your application to enable/disable in-flight features, allowing development (and testing) on trunk
simple

c::if test="${featureFoo}" turn it off in
   <a href="/foo">Foo</a> the user interface
</c::if>

public void doSomething() {
    if (featureFoo) {
        "foo specific logic"
    } else { turn it off in
        "regular logic" code
    }
}
inheritance

```java
public interface Processor {
    void process(Bar bar);
}

public class CoreProcessor implements Processor {
    public void process(Bar bar) {
        doSomething(bar);
        handleFoo(bar);
        doSomethingElse(bar);
    }

    protected void handleFoo(Bar bar) {
    }
}

public class FooProcessor extends CoreProcessor {
    protected void handleFoo(Bar bar) {
        doSomethingFooSpecific(bar);
    }
}
```
public interface FeatureHandler {
    void handle(Bar bar);
}

public class Processor {
    FeatureHandler handler;

    public Processor(FeatureHandler handler) {
        this.handler = handler;
    }

    public void process(Bar bar) {
        doSomething();
        handler.handle(bar);
        doSomethingElse();
    }
}

public class CoreHandler implements Handler {
    public void handle(Bar bar) {
    }
}

public class FooHandler implements Handler {
    public void handle(Bar bar) {
        doSomethingCompletelyDifferent(bar);
    }
}
annotations

@Retention(RetentionPolicy.RUNTIME)
public @interface Foo {
    boolean value() default true;
}

@Foo(false) public class CoreProcessor implements Processor {
    
}
@Foo public class FooProcessor extends CoreProcessor {
    
}
public class FeatureIncludeFilter implements TypeFilter {

    private final TypeFilter fooFilter = new AnnotationTypeFilter(Foo.class, true);

    public boolean match(MetadataReader metadataReader,
                          MetadataReaderFactory metadataReaderFactory)
            throws IOException {

        if (fooFilter.match(metadataReader, metadataReaderFactory)) {
            boolean value = getAnnotationValue(metadataReader, Foo.class);

            if (FeatureToggles.isFooEnabled()) {
                return value;
            } else {
                return !value;
            }
        }

        return false;
    }

    private boolean getAnnotationValue(MetadataReader metadataReader,
                                        Class annotationClass) {

        return (Boolean) metadataReader.
                getAnnotationMetadata().
                getAnnotationAttributes(annotationClass.getName()).
                get("value");
    }
}
<context:component-scan base-package="com.example.features">
  <context:include-filter type="custom"
    expression="com.example.features.FeatureIncludeFilter" />
</context:component-scan>

public interface Processor {

}

@Foo(false)
public class CoreProcessor implements Processor {

}

@Foo
public class FooProcessor extends CoreProcessor {

}
separating static assets

leave static assets as static files

create feature-specific versions of the static content

include conditionally into dynamic templates

shopping_cart.css

shopping_cart_foo.css
build vs runtime

build-time toggles:
  never leak details
  builds only what’s released

run-time toggles:
  long-lived feature toggles
  more flexible testing
cleaning up

remove feature toggles once feature becomes official

exception: multiple versions

don’t featurize your application to death
continuous delivery

When playing against someone who asks you to remind them about the rules, take the opportunity to subtly "suggest a throw" as you explain to them by physically showing them the throw you want them to play.
continuous integration
integrate early & often

deployment
deploy as the final stage of CI

delivery
software is always deployable
principles

create a repeatable, reliable process for releases

automate almost everything

keep everything in version control

if it hurts, do it more frequently

“done” means “released”
deployment pipelines
basic deployment pipeline

Source code

Env & app config

Version control

Commit stage
- Compile
- Commit tests
- Assemble
- Code analysis

Acceptance stage
- Configure environment
- Deploy binaries
- Smoke test
- Acceptance tests

Testers
Self-service deployments

Capacity stage
- Configure environment
- Deploy binaries
- Smoke test
- Run capacity tests

UAT
- Configure environment
- Deploy binaries
- Smoke test

Production
- Configure environment
- Deploy binaries
- Smoke test

Operations
- perform push-button releases

Artifacts repository

Developers
See code metrics and test failures

Reports
binaries metadata

bins

Reports
metadata binaries

Ops

Smoke test

Smoke test

Smoke test

Smoke test
<table>
<thead>
<tr>
<th>Revision</th>
<th>Resolution</th>
<th>Author(s)</th>
<th>Date</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0.0.5125</td>
<td></td>
<td>bdc7f35f9bc0... Jake, RRR, JJ, PS, Yogi, Anush</td>
<td>about 6 hours ago</td>
<td>reverted the confirmation popup added for pipeline trigger in pipeline activity</td>
</tr>
<tr>
<td>2.0.0.5124</td>
<td></td>
<td>25fcf692d54... ShilpaG &amp; Jake</td>
<td>1 day ago</td>
<td>reverted the confirmation popup added for pipeline trigger in pipeline activity</td>
</tr>
<tr>
<td>2.0.0.5122</td>
<td></td>
<td>2b006920224... ShilpaG &amp; Jake</td>
<td>1 day ago</td>
<td>reverted the confirmation popup added for pipeline trigger in pipeline activity</td>
</tr>
<tr>
<td>2.0.0.5121</td>
<td></td>
<td>15a21097f8d6... Yogi, PS</td>
<td>4 days ago</td>
<td>reverted the confirmation popup added for pipeline trigger in pipeline activity</td>
</tr>
</tbody>
</table>
DVCS magic
centralized VCS
DÉ-centralized VCS

commit
diff

pull
push

central repository
git magic #1
svn workflow

finish feature

svn up

run tests locally

kick off checkin bash script

wait...
continuous integration server

1. pull from svn
2. run local tests
3. check in

10 min / pair / check-in
continuous integration server

1. spawn local branch
2. pull from server
3. run tests
4. check in OR stash
5. kill branch
git magic #2
git magic #2

git server

1. undo disastrous checkout
2. save changes to local stash
3. create local branch
4. push stash to local branch
git magic #2

5. push local branch to remote branch

6. you broke it — you fix it!
git magic #2

6. stash recent changes
7. checkout remote branch
8. fix it!
9. check into main
10. unstash & get back to work
transfer a merge conflict to the person better qualified to fix it.
why all the rochambeau?
view builds

01_trunk_commit 02_trunk_acceptance 03_trunk_apache
04_trunk_externals 05_trunk_metrics 07_trunk_qa_tests
11_release_commit 12_release_acceptance 13_release_apache
14_release_externals 17_release_qa_tests 97_deploy_ba
98_deploy_staging 99_spider_production ove-search-infrastructure
in-service ove-core-trunk ove-core-release ove-datasets
ove-externals ove-externals-trunk ove-query-counts
webservices-core z-deploy-ba-trunk z-deploy-endeca-ba-trunk
z-deploy-iqa-release z-deploy-sqa-trunk ove-view-trunk
ove-view-release-branch
worst ...job ...ever
please fill out the session evaluations
resources

XProgramming.com – Ron Jeffries site
http://xprogramming.com/

Dr. Laurie Williams
http://collaboration.csc.ncsu.edu/laurie/publications.html

git branching model
http://nvie.com/git-model

Extreme Programming: A Gentle Introduction
http://www.extremeprogramming.org/
Maintain a Single Source Repository.
Automate the Build
Make Your Build Self-Testing
Everyone Commits To the Mainline Every Day
Every Commit Should Build the Mainline on an Integration Machine
Keep the Build Fast
Test in a Clone of the Production Environment
Make it Easy for Anyone to Get the Latest Executable
Everyone can see what's happening
Automate Deployment

http://martinfowler.com/articles/continuousIntegration.html