relevance

iteration zero

stuart halloway
http://thinkrelevance.com

how relevance does izero

may not work

for you!

iterations are the heartbeat

regular rhythm

opportunities for feedback and adjustment

proof (and measurement) of progress

built-in stopping points

good iterations

deliver business value

build features, not layers

end with a deployable system

“that first step’s a doozy!”

architecture

infrastructure

discovery

teaching

have a checklist

Select: All | None Listed below: 1 to 15 of 15.

	▲ #	Name	Priority	Status
☐	1	Set up Git repository		Open
☐	2	Check-in project skeleton		Open
☐	3	Set up staging server		Open
☐	4	Set up automated deployment		Open
☐	5	Set up default rake task with tests, complexity analysis, coverage threshold		Open
☐	6	Set up continuous integration		Open
☐	7	Set up tarantula		Open
☐	8	Mingle/story life-cycle demo		Open
☐	9	Establish standup schedule		Open
☐	10	Choose and setup database for the project		Open
☐	11	Provision conference line		Open
☐	12	Enter member contact information		Open
☐	13	Meeting to go over process/meetings		Open
☐	14	Set up basecamp		Open
☐	15	Describe deployment environment for first production release		Open

the initial wag

can be bottom up, top down, or both

use points, not hours

give a range, not a number

explain the key variables

prioritize time/money/features

I. value individuals and interactions over processes and tools

individuals

identify team roles

choose the right people for those roles

roles

sponsor

stakeholders

project manager

business analysis

development

quality assurance

and more...

plan interactions

daily standups

card transitions

iteration reviews

iteration planning

retrospectives

risk analysis

processes and tools

mailing lists

card tracking

wiki

chat room

pairing

remote pairing

2. value working software over comprehensive documentation

working software

is *deployed* software

get there quickly, and often

snapshot

keep working software

testing infrastructure

testing feedback loops

start with a clean build

keep it that way

comprehensive documentation?

good agile gives

great

documentation!

things worth documenting

policies and plans

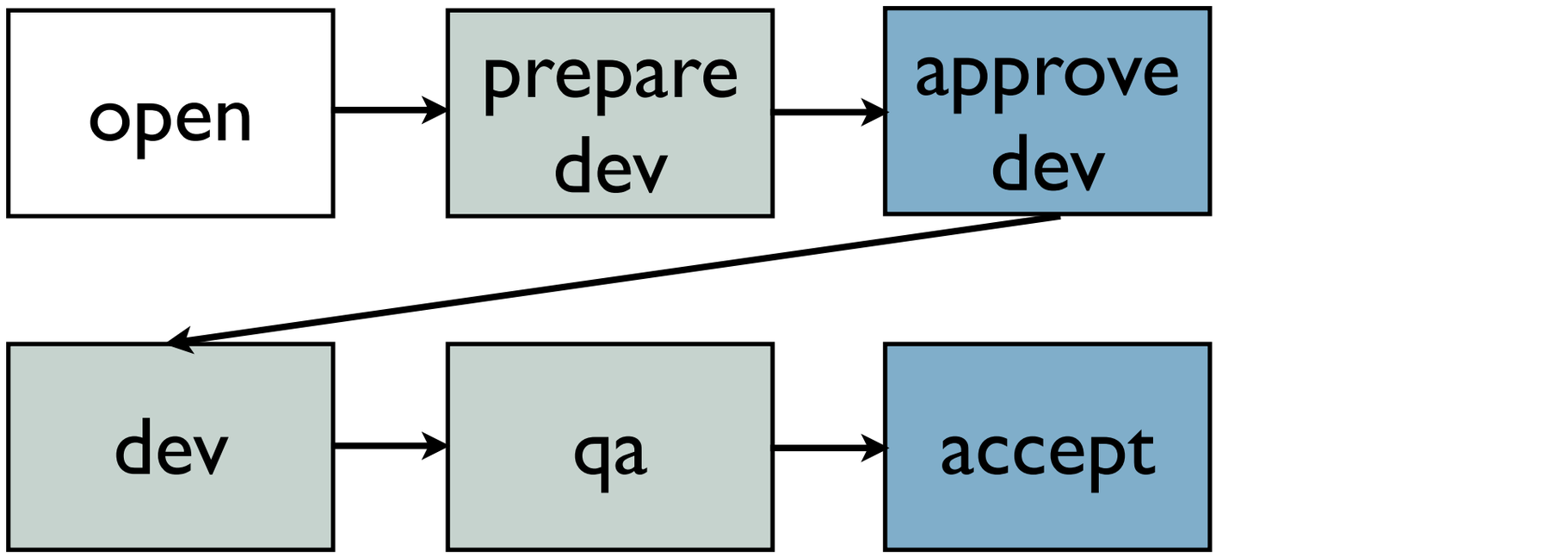
deployment

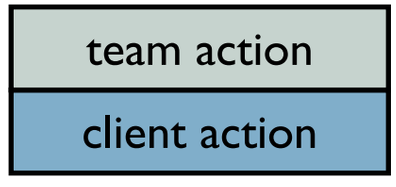
architecture (partially implicit)

card transitions!!

3. value customer collaboration over contract negotiation

a flow that works





everything

is a card

stakeholder control knobs

creating cards

approving cards for development

accepting cards

raising issues (standup)

escalating issues

stop at any time (with working software)

bugs?

preventing bugs

card transitions

objective acceptance criteria

pair programming

test-driven development

continuous integration

bugs will

still happen

responding to bugs

write a test

fix the problem

did the people fail?

did the process fail?

is it worth preventing in the future?

fixing bugs

is new work

4. value responding to change over following a plan

changing priorities

create new stories

reorder the backlog

deploy when needed

technical spikes

estimation spikes

escalation paths

individual -> standup -> iteration planning

retrospectives

risk assessments

relevance

questions?

stuart halloway
http://thinkrelevance.com