Continuous Integration

### Software development life cycle for Force.com projects

April 9th 2015

Aldo Fernandez



Technical Consultant

@aldoforce

Safe Harbor

Safe harbor statement under the Private Securities Litigation Reform Act of 1995: This presentation may contain forward-looking statements that involve risks, uncertainties, and assumptions. If any such uncertainties materialize or if any of the assumptions proves incorrect, the results of salesforce.com, inc. could differ materially from the results expressed or implied by the forward-looking statements we make. All statements other than statements of historical fact could be deemed forward-looking, including any projections of subscriber growth, earnings, revenues, or other financial items and any statements regarding strategies or plans of management for future operations, statements of belief, any statements concerning new, planned, or upgraded services or technology developments and customer contracts or use of our services.

The risks and uncertainties referred to above include – but are not limited to – risks associated with developing and delivering new functionality for our service, our new business model, our past operating losses, possible fluctuations in our operating results and rate of growth, interruptions or delays in our Web hosting, breach of our security measures, risks associated with possible mergers and acquisitions, the immature market in which we operate, our relatively limited operating history, our ability to expand, retain, and motivate our employees and manage our growth, new releases of our service and successful customer deployment, our limited history reselling non-salesforce.com products, and utilization and selling to larger enterprise customers.

Further information on potential factors that could affect the financial results of salesforce.com, inc. is included in our annual report on Form 10-K for the most recent fiscal quarter ended July 31, 2011. This document and others are available on the SEC Filings section of the Investor Information section of our Web site.

Any unreleased services or features referenced in this or other press releases or public statements are not currently available and may not be delivered on time or at all. Customers who purchase our services should make the purchase decisions based upon features that are currently available.

Salesforce.com, inc. assumes no obligation and does not intend to update these forward-looking statements.



# Follow UYSDUG for the latest news

### @uysdug #uysdug

 /Uruguay-Salesforce-Developer-User-Group/ [www.uysdug.com](http://www.uysdug.com/)



Join us and share your knowledge!

* Got something to share with the world?
* We are open for talks/sessions
* Check [www.uysdug.com/cfp](http://www.uysdug.com/cfp) or get in contact with someone from the Technical Board for more information
* DF15 call for presentations will open on April’15
* UPCOMING EVENTS:
  + May (Javascript)
  + June (Analytics Cloud / Wave)\*
  + July (Data Integration)\*

Agenda

* What is CI?
* Different Force.com scenarios
* Story time
* Theory time
* Approach to CI
* Approach example
* Benefits
* Best Practices
* Consideratons



Continuous Integration

Also known as:

* Continuous Deployment
* A discipline that allows developers to enjoy Friday’s nights (instead of late overnight shifts because build errors)



Continuous Integration

“Is a software development **practice** where members of a team integrate their work **frequently**. Each integration is verified by an **automated build,** including **test**, to detect integration errors as quickly as possible”

—Martin Fowler



Force.com scenarios

* You are an **ISV Partner** working on a product that will end on AppExchange (as a **Managed Released Package**)
* You are a **company using Salesforce** as a key part of your business and **business requirements** need implementation.
* In both scenarios you have a **team** (with different **roles**) and a group of stakeholders
* We are going to study the “Company using Salesforce” scenario



Story time (based on real facts)

**barkingtshirts.com**



* Promising startup
* Got some VC $
* By suggestion acquired Salesforce EE org licenses
* Initial requirements: CRM standard usage
* no… don’t google it, it’s fiction



barkingtshirts.com - Early years



Production

Tom (CEO)

Production

Sandbox

Sales Team

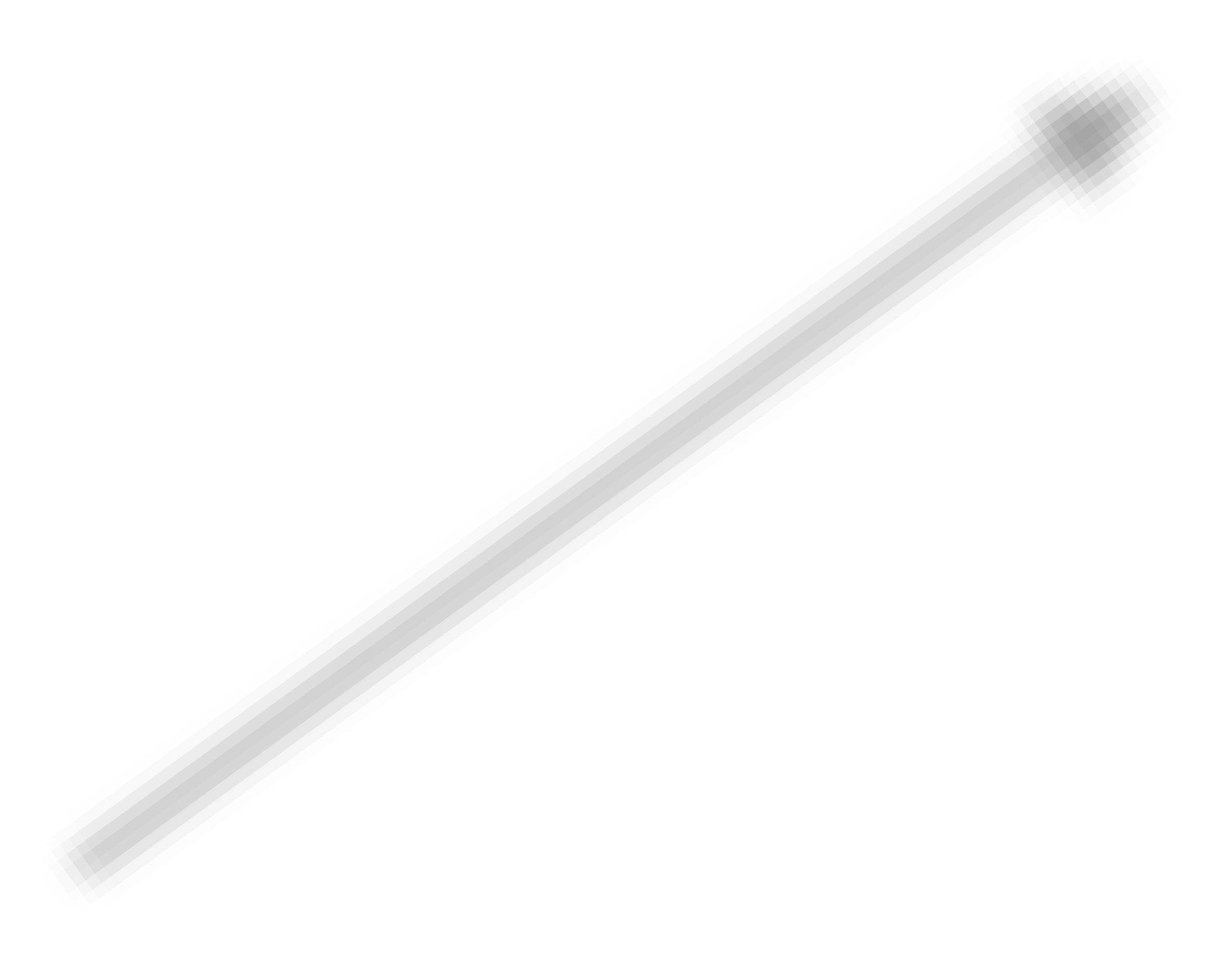
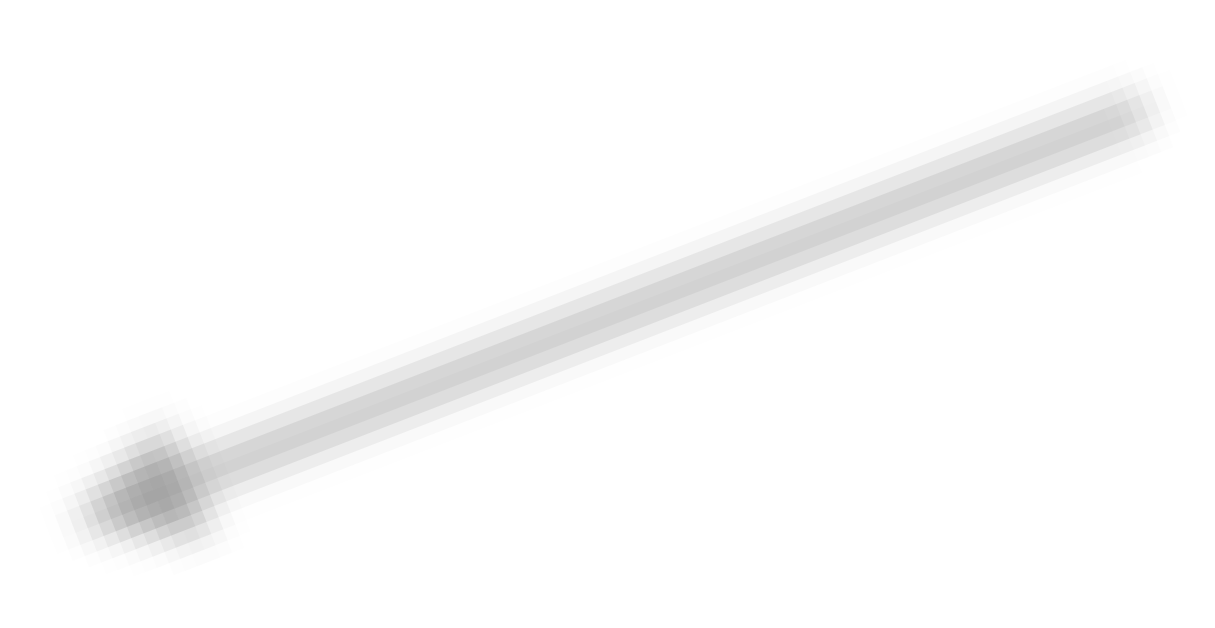
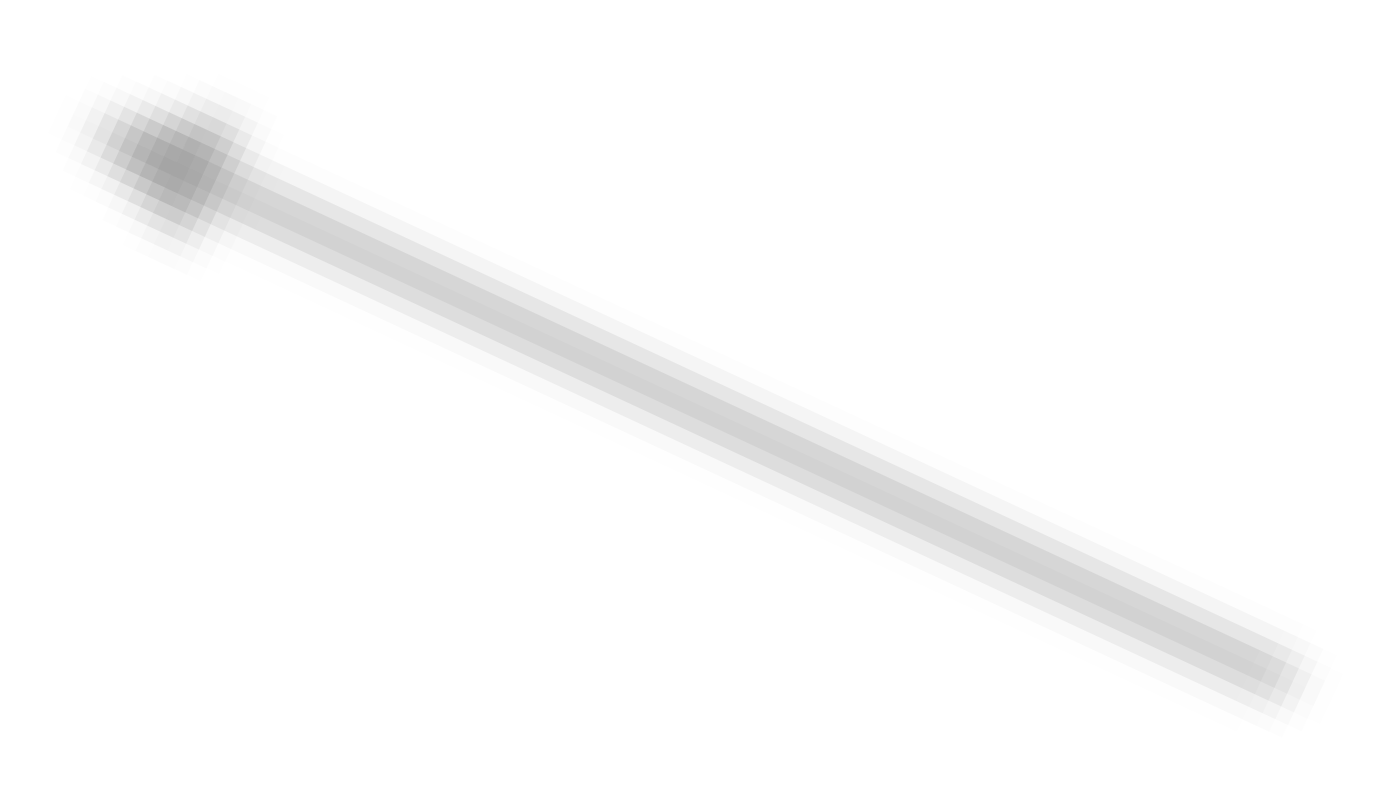
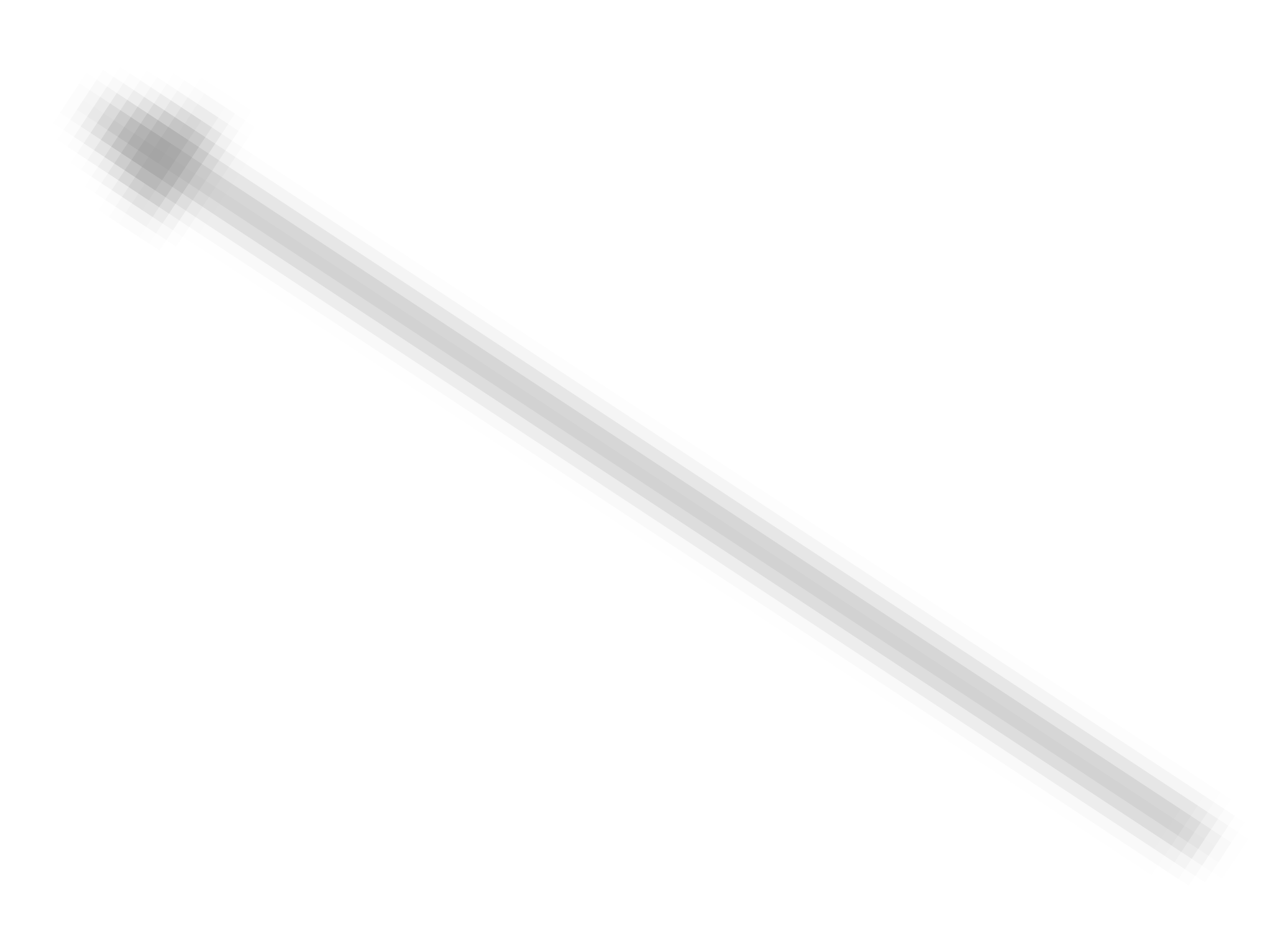
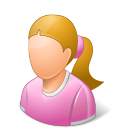
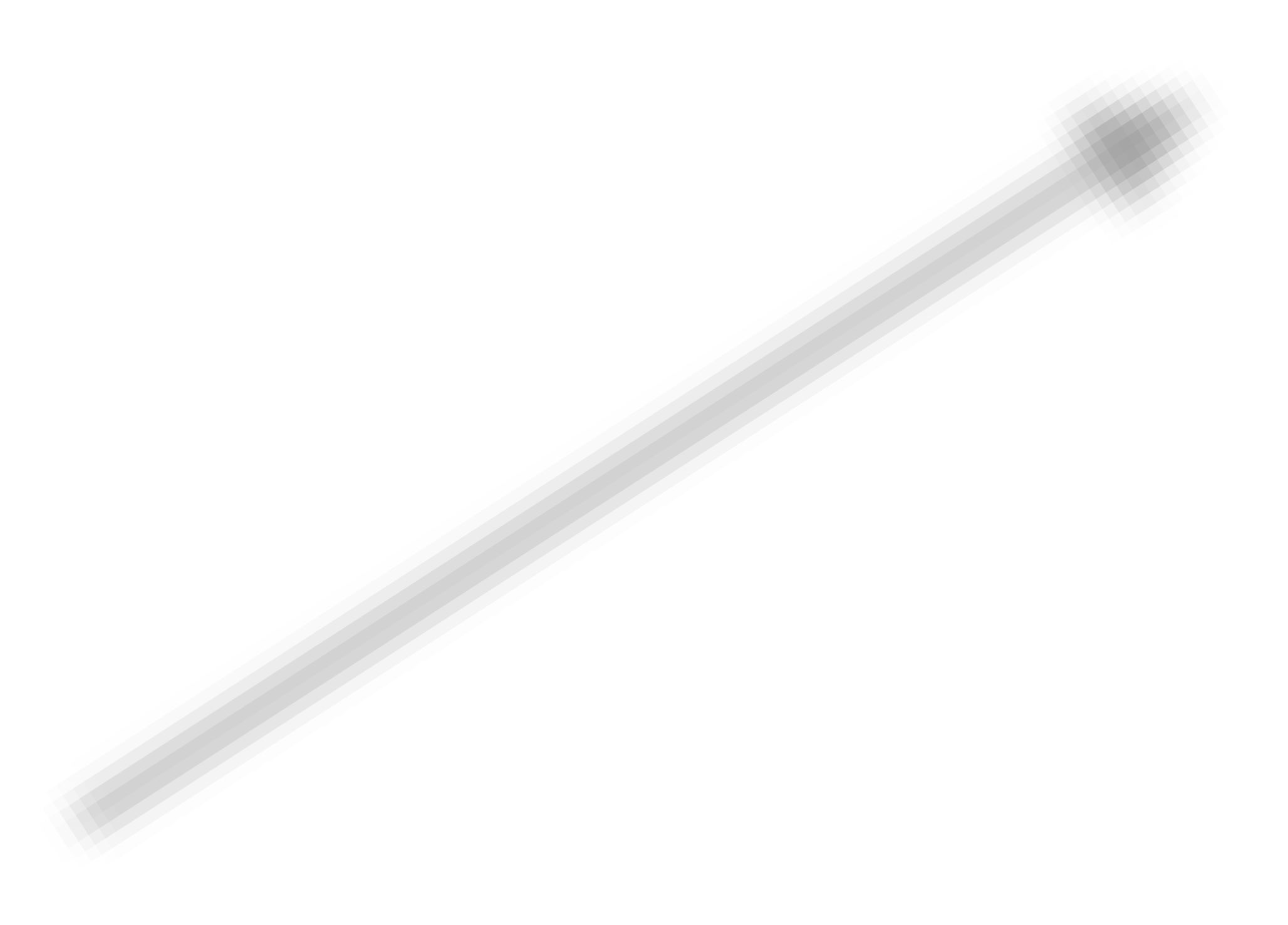
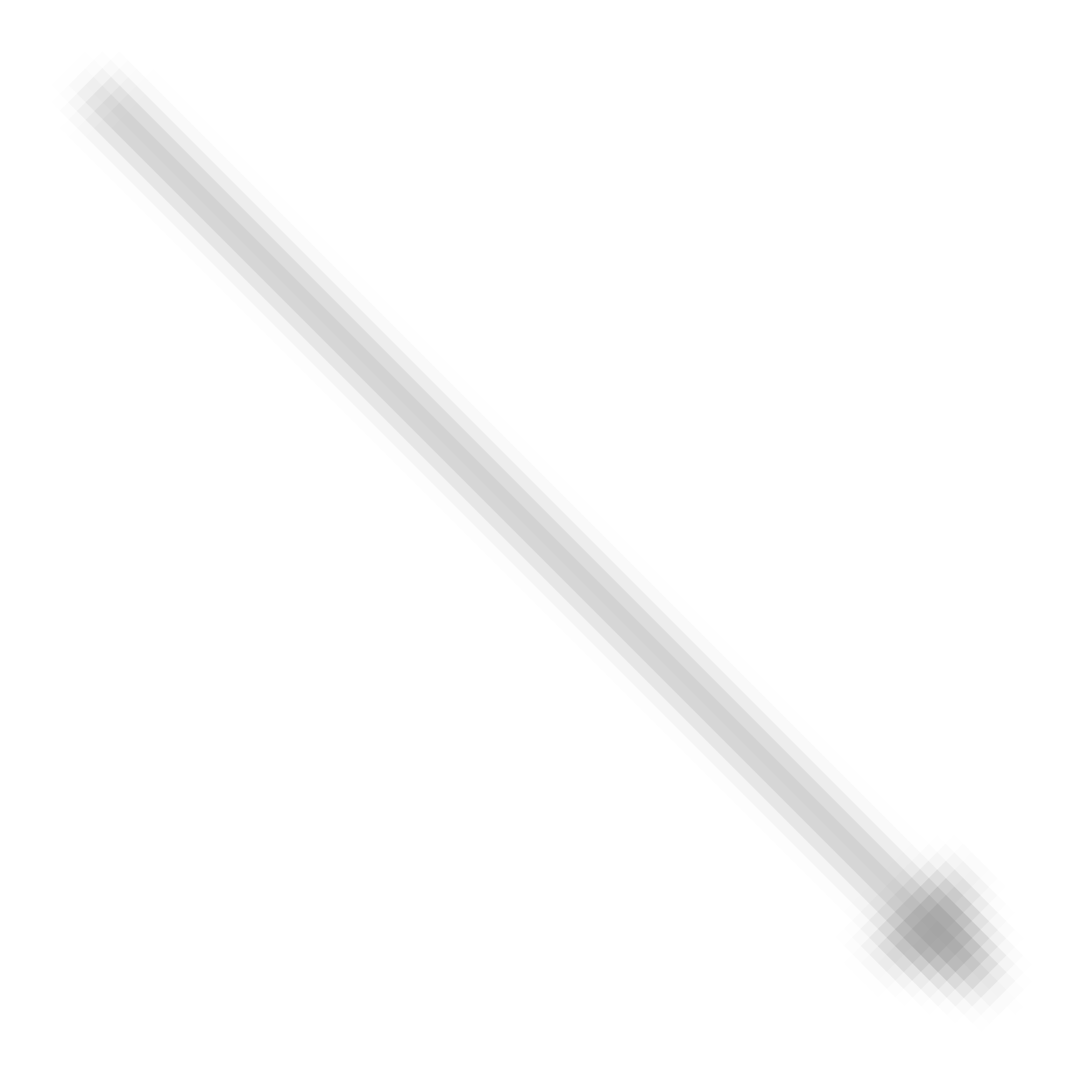
Olaf (Admin)

Sales Team Trainees



Fred (Developer)

Chad (Mobile Dev)



Ruth (Developer)



barkingtshirts.com - Early years - issues

#### Business and team grow quickly

* + Infrastructure and ORG layout inadequate
  + Too many hands on the same code base
    - changes get overwritten by other devs
    - conflicts errors discovered late in process
    - limited visibility/audit of changes
  + Critical business impact

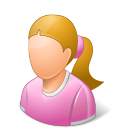
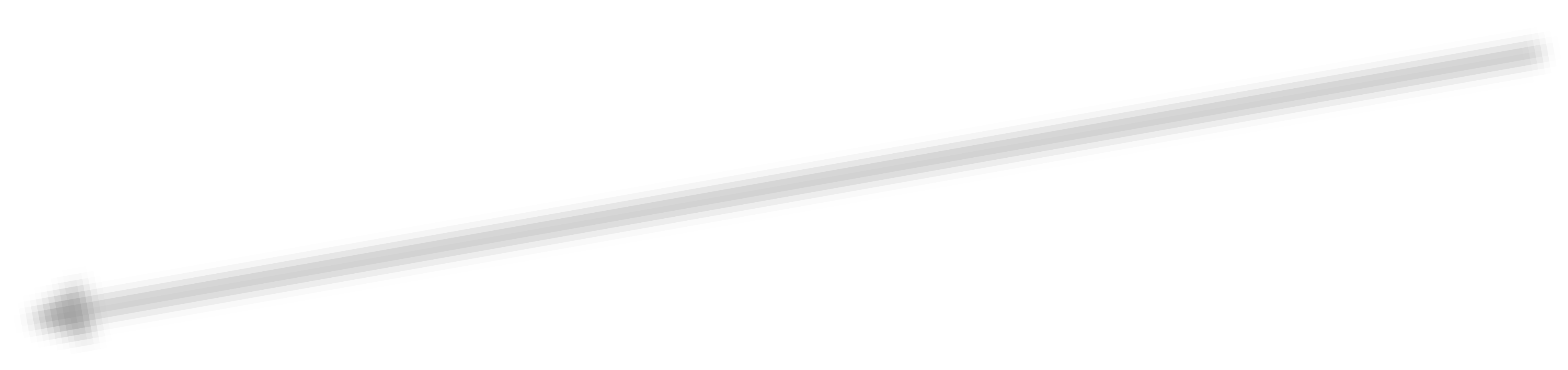
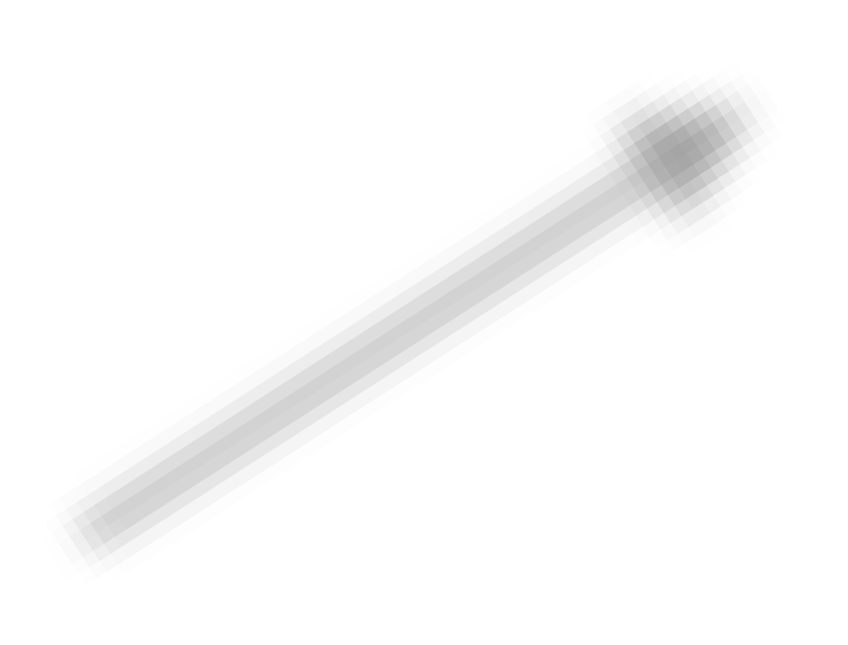


Separate environments for different roles!

* + Production
  + Staging (UAT)
  + QA
  + Training
  + Development
  + Config



barkingtshirts.com - After some planning



Sales

Team

Sandbox (QA)

Sales Team

Trainees

Sandbox (Dev2)

Sandbox (Dev3)

Sandbox (Dev1)

Sandbox (Config)

Sandbox (Training)

Sandbox (Staging)

Production

Tom (CEO)



Olaf (Admin)

Fred (Developer)

Ruth (Developer)

Chad (Mobile Dev)

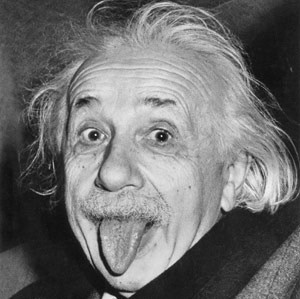


## barkingtshirts.com - After some planning - issues

#### Keep orgs in sync as a team

* + Changes are too slow
  + No visibility of change’s traceability
  + Manual or NO merging at all



**Theory time!**



## How to merge code?

#### Using Metadata API to get all codebase (package.xml)

* + Source code control system
    - Git
    - SVN
    - SourceSafe
    - etc



How to move changes between orgs?

* + Salesforce ANT migration tool
  + Force.com IDE / Eclipse + Force.com plugin
  + Change sets
  + Manual changes at target org(\*)



How to automate builds?

* + Jenkins
  + Bamboo
  + Cruise Control
  + TravisCI
  + Aegis
  + Codeship
  + JetBrains
  + …



Approach - ORGs layout

Production

Hot fix

Staging



Training

Sandbox Org type:

UAT

Test execution



Live

Full Sandbox

Partial Copy

Developer Pro

Developer

QA

CI

###### Jenkins

Git repo



Config

Dev1

Dev2

Dev3



## Approach - Components

* + Salesforce ORGs for development
    - Sandboxes (different flavors)
    - DE Orgs
  + Git for source code management
    - Github
    - Bitbucket.org
    - Command line as client
  + Continuous Integration for automated builds
    - Jenkins or Bamboo for orchestration
    - Ant Force.com Migration Toolkit for deployments



Approach - Components - Salesforce ORGs

* + Separate orgs for each developer
  + Dedicated CI org
    - where all code will be integrated
    - all tests will be executed
  + Different orgs for each life cycle stages
    - QA
    - UAT
    - Staging
    - Training
    - Hot fix (\*)



Approach - Components - Git

* + Store source code & metadata
  + Include libraries and scripts for deployment
  + Track changes
  + Isolate features and stages of the life cycle
    - branching for features or sprints
    - tagging for releases and versioning
  + Mock data for configuration and fixtures



Approach - Components - CI Jenkins

###### Automation

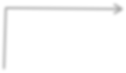
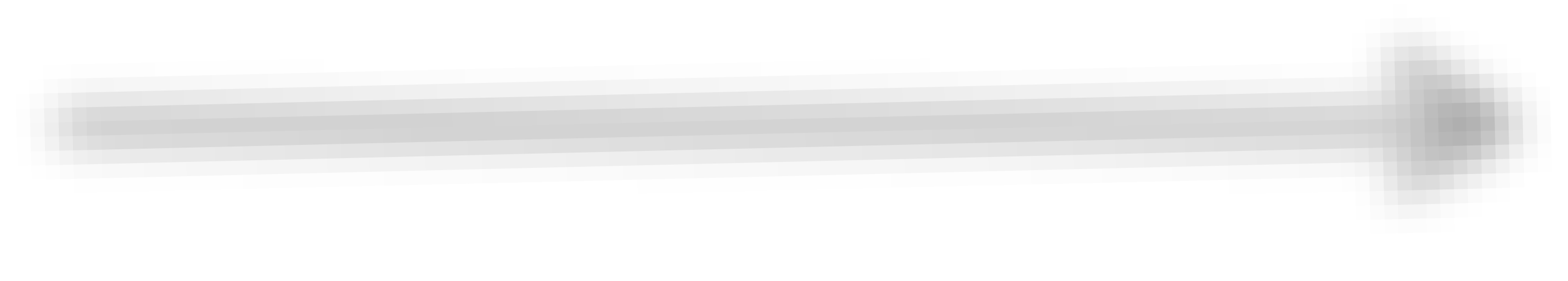
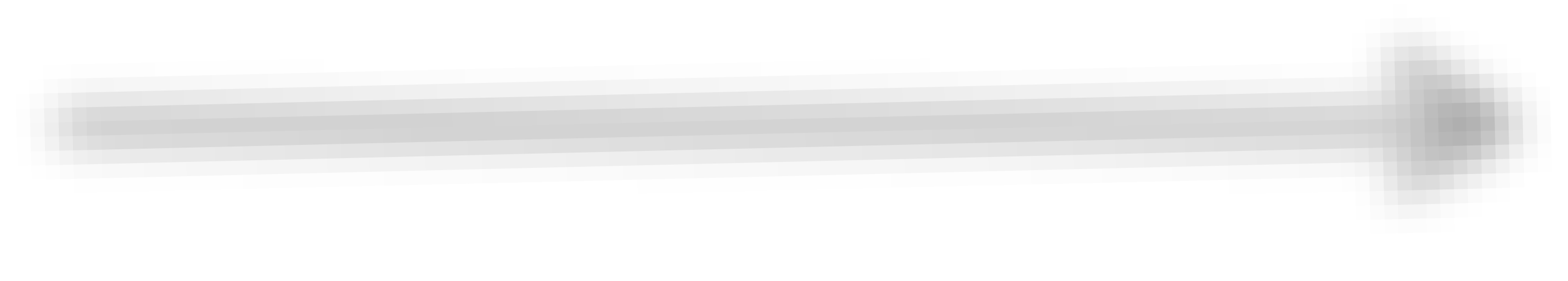
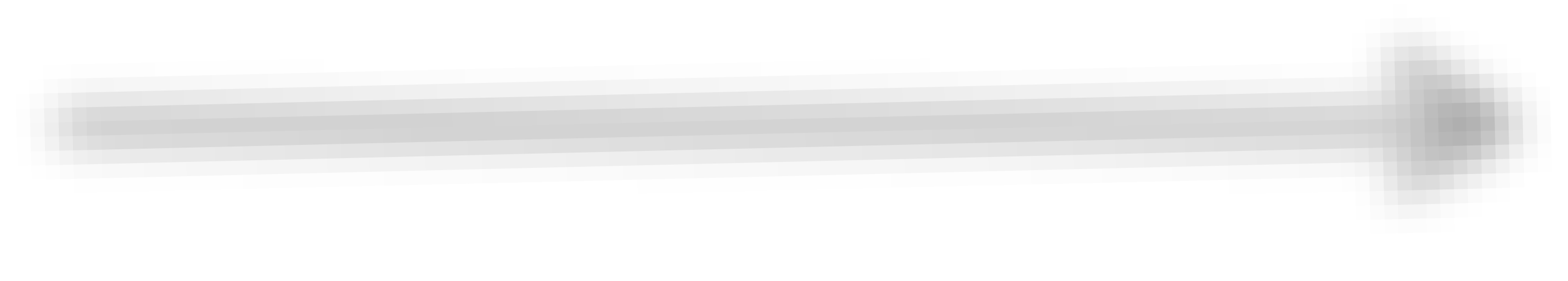
* + - Ant based deployments from git repo
    - Execution of supporting tasks
    - Notifications
      * merge conflicts
      * build/test errors
    - Reporting
  + Setup
    - data migration (configuration/fixture tests)
    - execution of setup scripts
  + Testing
    - Unit test execution
    - UI test execution (Integration with external apps)



Approach - An example of a daily dev task



Fred (Senior Developer)



1. if ok deploy to other orgs
   1. ok to merge

QA

* 1. merge and push

###### Git repo

1. fetch

* 1. start build

5. pull request

###### Jenkins

1. deploy to CI
2. execute tests

CI

Test execution

Ruth (Developer)

Local computer

* 1. push to org
  2. makes updates

Dev

* 1. success!

## Approach - Benefits

* + Source code backup
  + Automation
    - automated deployments to multiple orgs
    - automated test execution
    - reduce human error factor from deployment
  + Audit and changes visibility
  + Validation
    - issues get detected early in the process
    - resolve issues close to cause (less time debugging)



Approach - Best practices

* + Start simple and KISS
  + Automate as much as possible
  + Integrate with your ticketing system
  + Use git branching and tags
  + Pull requests to merge features
  + Commit and build regularly



Some considerations about automated deployments

* + You can’t automate everything :-(
    - Metadata can get/set 90% of components (Classes, SObjects, layouts)
    - Account Teams, Case Team, Console layout, etc. need manual deploy
  + Large deployments can take a lot of time, a single test failure may requires aborting the whole process
  + Some changes might take too much time: sharing rules, inefficient tests, processing time
  + Platform and servers availability (avoid peak-time)
  + Can’t rollback a deploy (roll forward only)
  + You can deploy SObjects schema changes but cannot automate data changes



Resources + Kudos

##### Seb Wagner @se6wagner “Automating deployment between orgs using git”

* Martin Fowler @martinfowler
* Anup Jadhav @anup “Salesforce deployment thoughts”
* Jenkins jenkins-ci.org
* Atlassian - “Putting a lasso on Salesfoce Development Process”
* Jeff Douglas @jeffdonthemic “Setting up continuous integration for salesforce development”
* CumulusCI
* Kevin O’hara @kevohara “Setting up Jenkins for CI”
* Bracket labs “Appexchange Team Development on force.com platform”
* Kevin Bromer @kevinbromer “Force CI using AWS, Github and Jenkins”
* CloudBees - “Jenkins on the clouds”



Thank You!

#uysdug #CI

Uruguay Salesforce Developer Group

April 9, 2015 #UYSDUG

Continuous Integration

Software development lifecycle for Force.com projects

Aldo Fernandez Technical Consultant

@aldoforce