

# Integrating Accessibility in Rapid Development

Brian Cragun (presenter) [cragun@us.ibm.com](mailto:cragun@us.ibm.com)  
Susann Keohane [skeohane@us.ibm.com](mailto:skeohane@us.ibm.com)  
IBM Human Ability and Accessibility Center

February 27, 2013





- What are the differences between traditional software development and rapid development processes?
- *What do “agile”, “waterfall”, and “continuous development” mean for software developers and accessibility testers?*
- How can traditional accessibility processes be adapted for rapid development
- *What potential benefits do rapid process hold for accessibility?*
- How can you effect change in your rapid processes to benefit accessibility?



Photo credit: Flickr / Colin\_K



- Review common software development models
- Discuss where accessibility fits into each model
- Benefits and challenges
- Tooling
- Case studies and industry examples
- Adapting to other models
- Continuous delivery
- Approaching change in your enterprise
- Conclusion



Photo credit: Flickr/kevinzim



- We'll talk about how IBM is approaching accessibility in development processes with the goal of helping you improve the integration of accessibility into your processes.
  - This is not an announcement of products or services
- Companies, platforms and products mentioned hold rights to their respective trademarks
  - No endorsement of other products is inferred
  - See more trademark notices at the end
- Conclusions are based on anecdotal experience
  - There may be errors and omissions (and probably even a few wild assumptions)
- I am not a lawyer
  - So even these statements may not be adequate disclaimers.

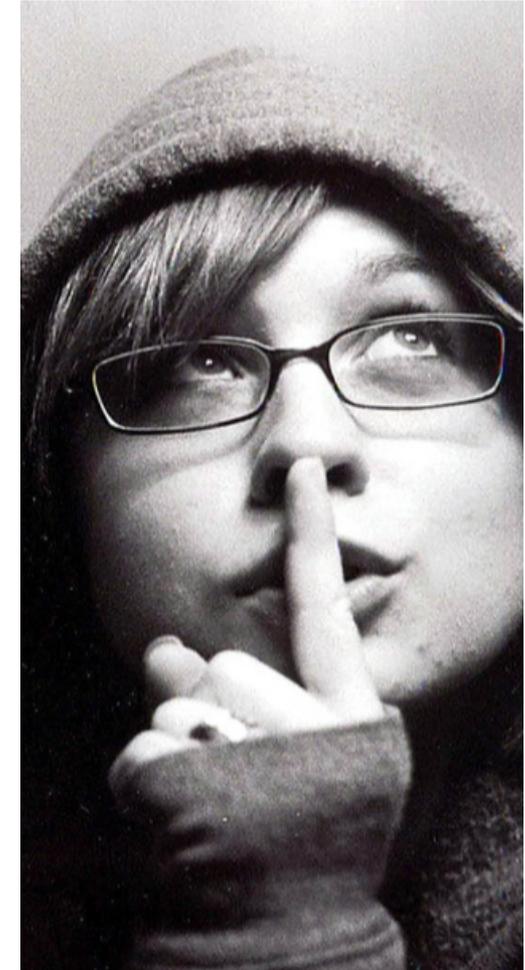
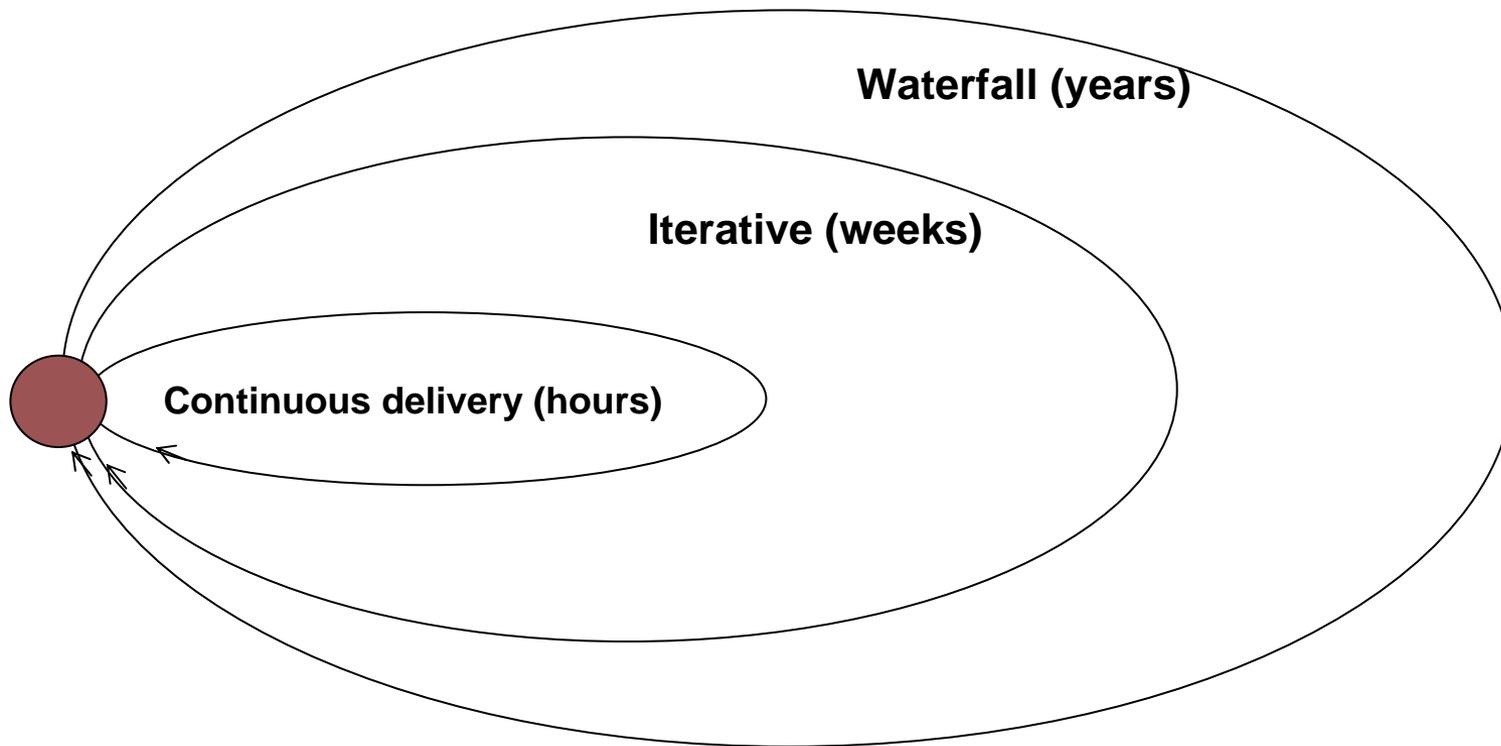


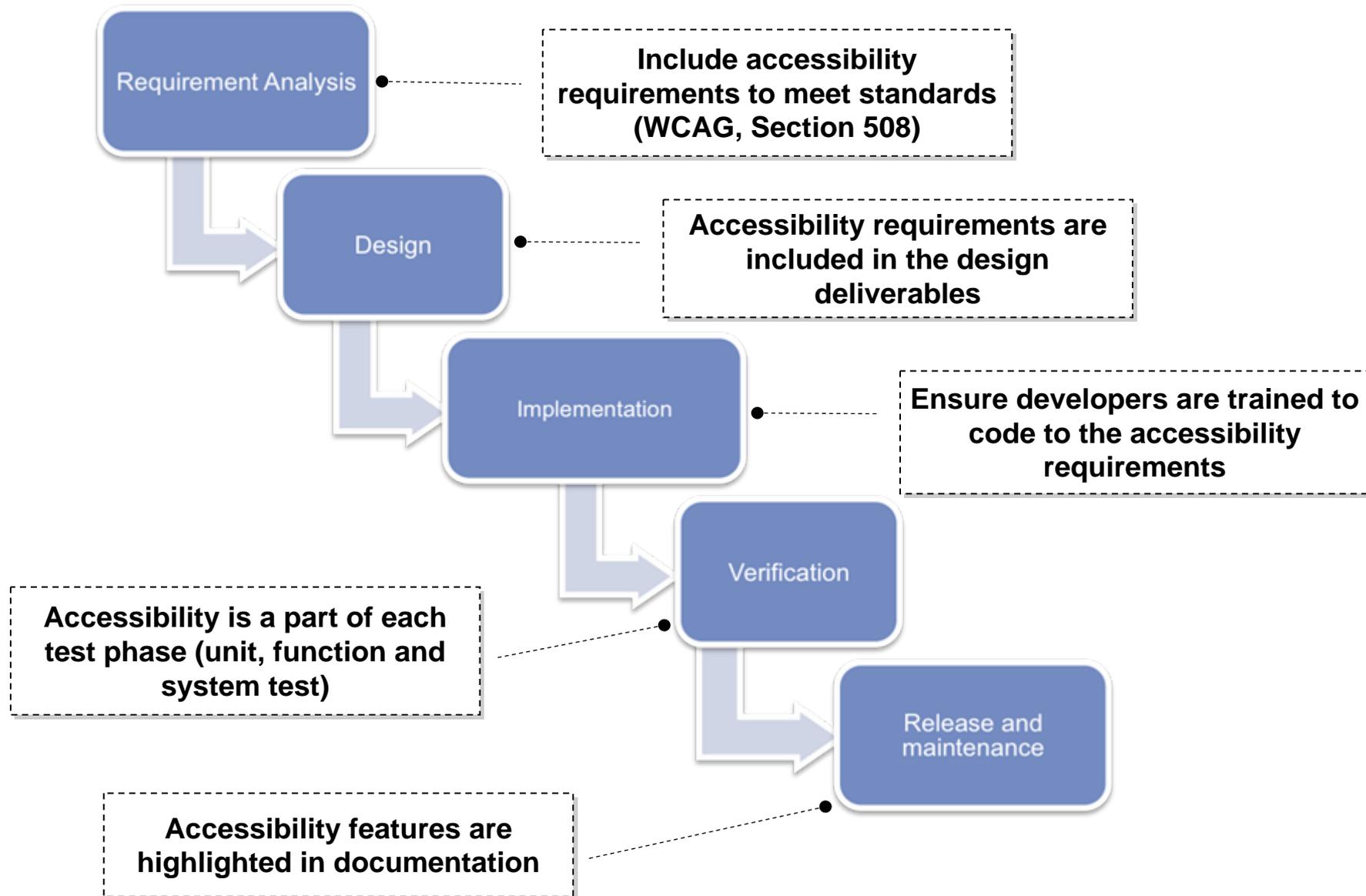
Photo credit: Flickr/Katie Tegtmeier



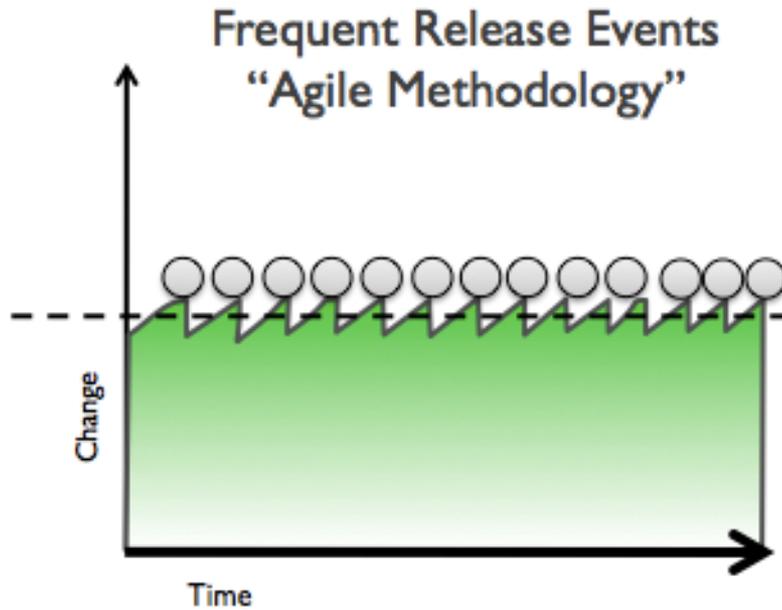
## Traditional development phases



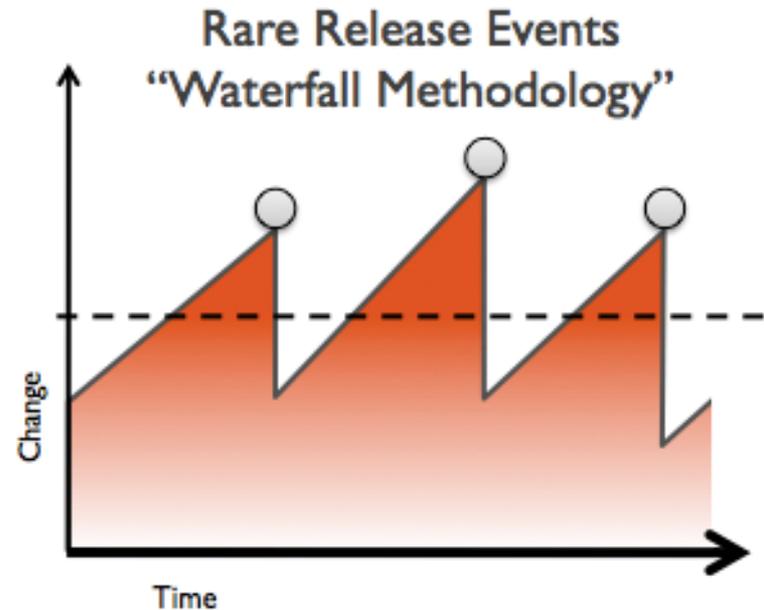
**Challenge is to include accessibility within shorten cycle**



# What is Agile methodology?



Smoother Effort  
Less Risk



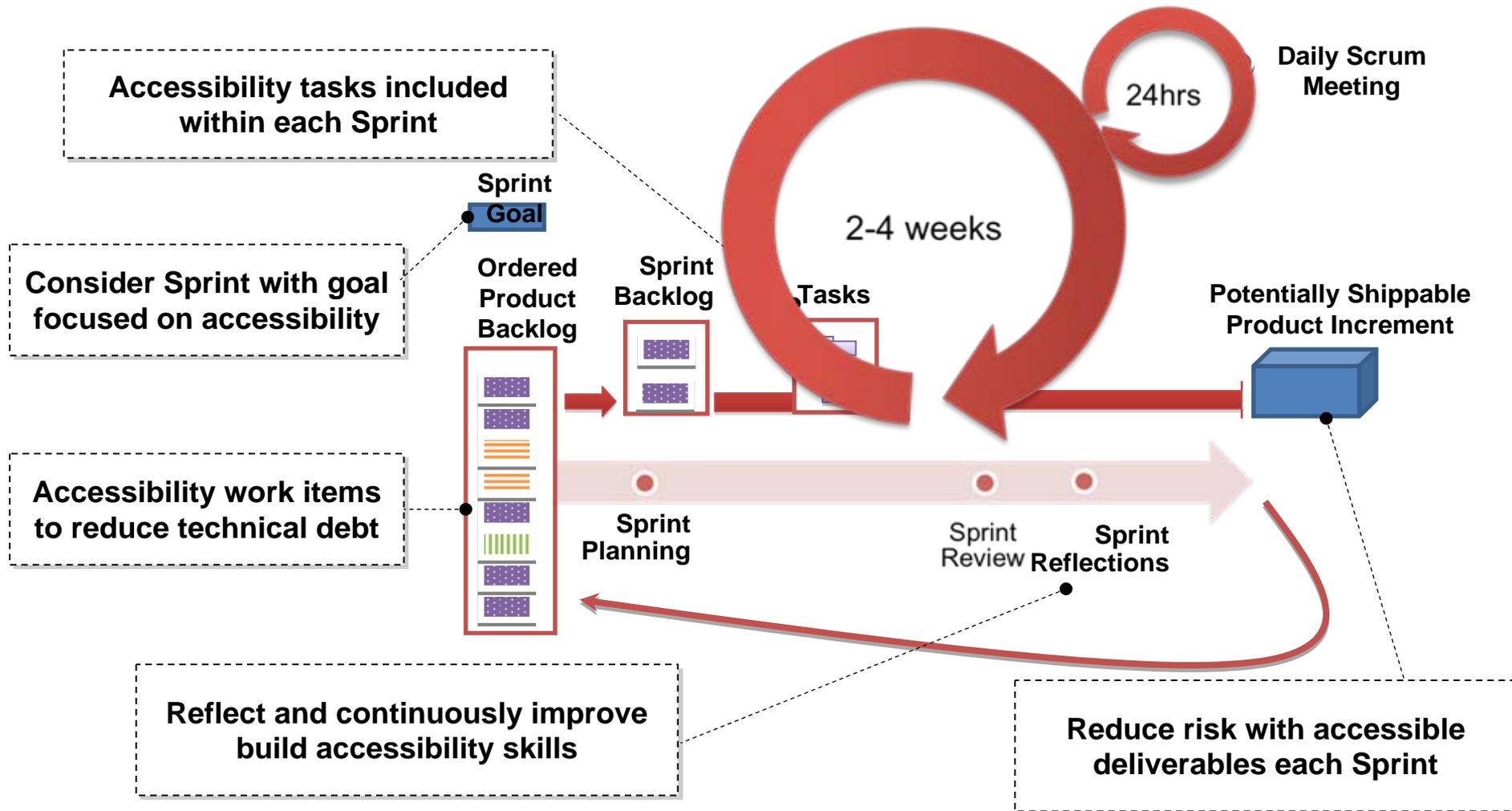
Effort Peaks  
High Risk



**How does accessibility fit into a more frequent release cycle?**



## Early Focus Reduces Cost, Risk





## Dedicated Accessibility Sprint(s)

### *Advantages:*

- Closer to waterfall paradigm (familiar)
- Focus on accessibility
- May save implementation time on accessibility
- Supports roving specialists for development and test
- Supports broad system testing for accessibility

### *Disadvantages*

- Adding accessibility later invariably requires rework.
- No feedback from PwDs in early sprints
- Design may be “locked down” by the time accessibility is added. Resistance to change and “undoing” elements already committed previous Sprints.

## Accessibility in Each Sprint

### *Advantages:*

- Useable features at each cycle
- Allows Stakeholder feedback to include accessibility
- Stakeholders can include PwD
- Supports rapid / continuous development
- Supports consistent team and test resources
- Capability of release at end of each Sprint

### *Disadvantages*

- Consistent set of expertise
- May take more total accessibility resource





## Tools enable faster process

- Automated testing
  - Find and correct problems when they are coded and checked in.
- Interactive testing and diagnostics
  - Where are the problems occurring
- Keep track of requirements and prioritize
  - Requirements management
  - Defect management

## IBM tools

- Rational Policy Tester
  - Find and correct problems when they are coded and checked in.
- Dynamic accessible Plugin (built on Firebug)
- Rational Collaboration Lifecycle Management
  - Uses ReqIF standard



# Align business, development and test efforts



## Working in a Development Lifecycle (ALM Today)

Requirements driven development across the lifecycle



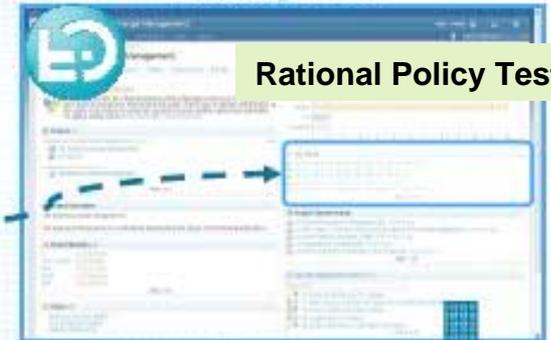
### Rational Requirements Composer 4.0



### Rational Team Concert 4.0



### Rational Quality Manager 4.0

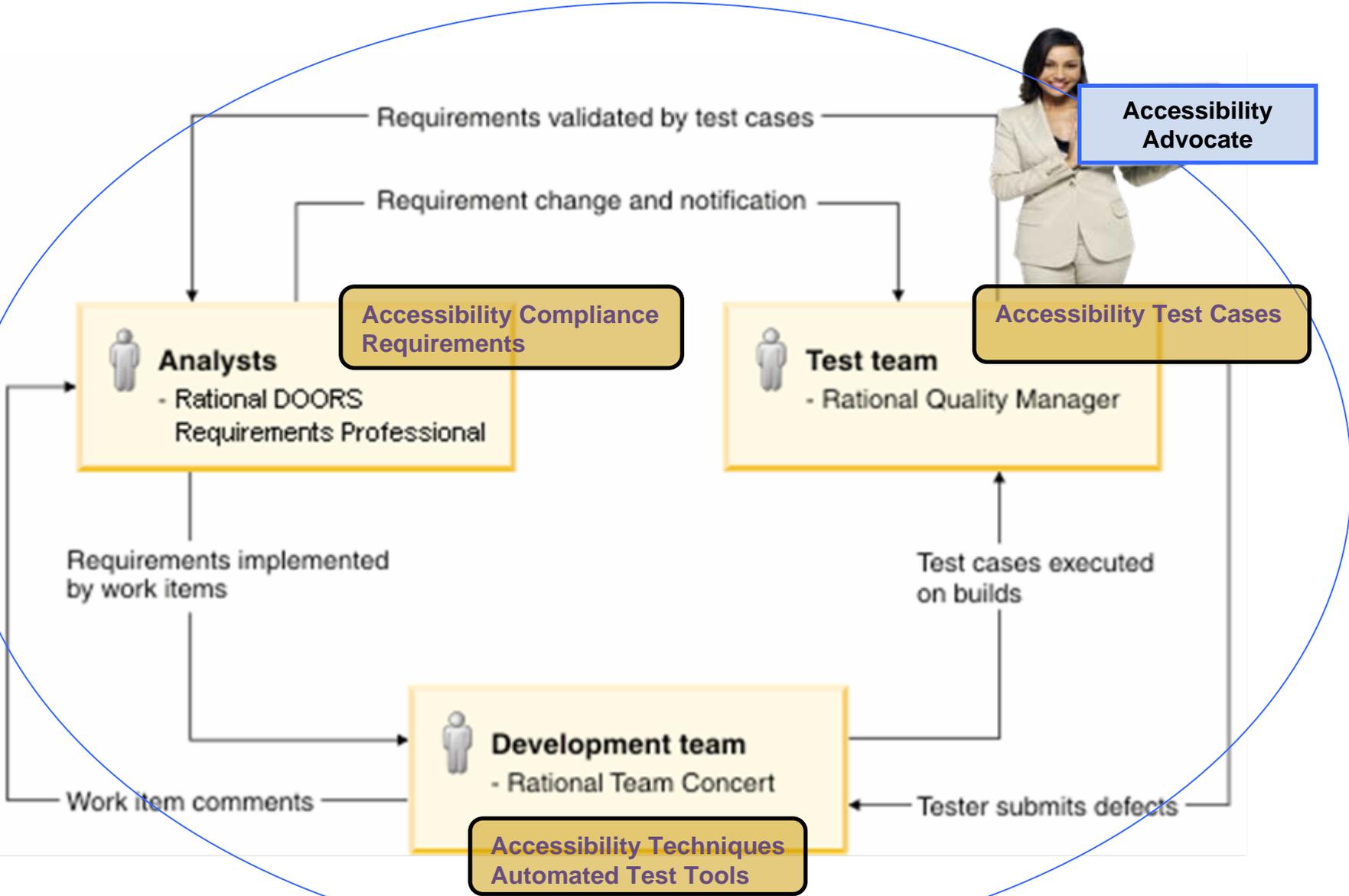


Rational Policy Tester

ID	Name	Artifact Type	Implemented By	Validated By
8	Dividend allocation by percentage	Feature	75: Dividend Allocation by Percentage	18: Dividend Allocation by Percentage
	Requirements	Feature	6: Customers can Nominate an organization	21: Customers can nominate an organization for the program
12	Donor Dividend Allocation Criteria	Feature	73: Donor Dividend Allocation Criteria	10: Customers can Nominate an Organization
	Satisfied by			16: Donor Dividend Allocation Criteria
23	Frequency of dividend transfer	Feature	58: Frequency of dividend transfer	1: Frequency of dividend transfer

Single View for End to End Development Needs

# Accessibility by role and responsibility





- Understands accessibility
  - Needs of PwD
  - Assistive technologies
  - Standards
- Understands architectures and processes
  - Software stack
  - User Interface (UI) technologies
  - Documentation
- Provides guidance
- Knows techniques
- Educates teams and management
- Watches over processes
- Suggests process application and improvement



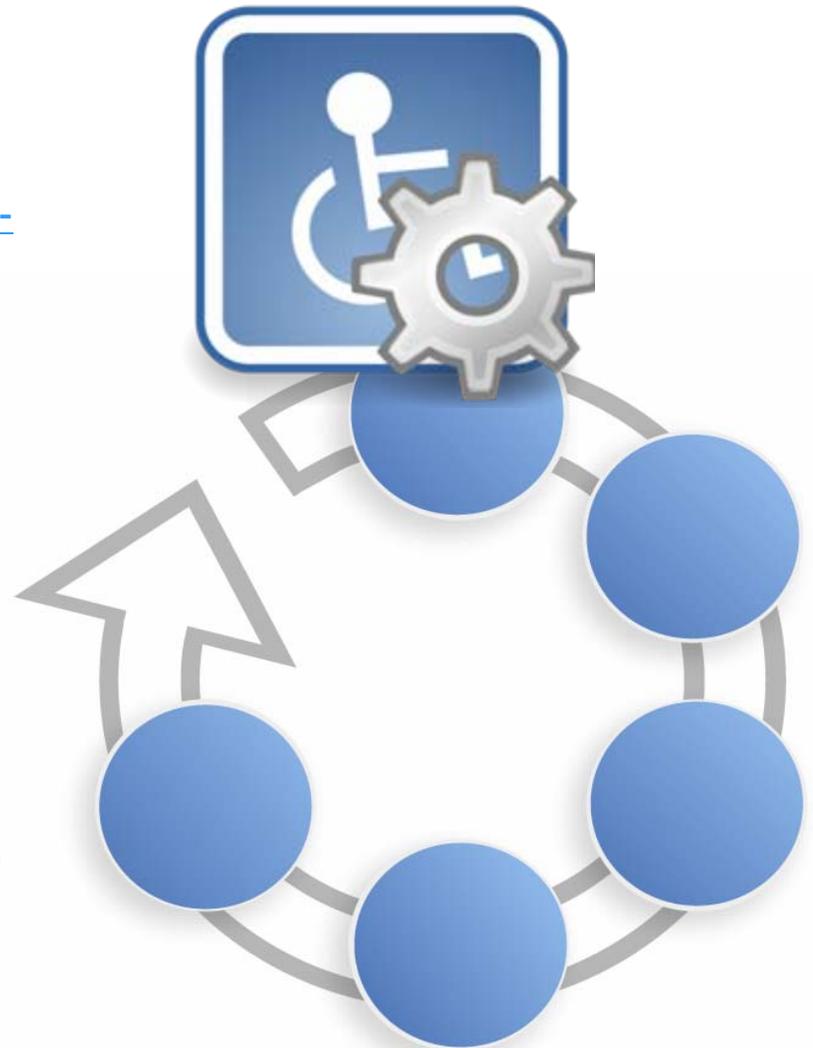


- Add accessibility to planning and design
- Teaming and co-location
- Early automated testing
- Rapid iteration and stakeholder feedback



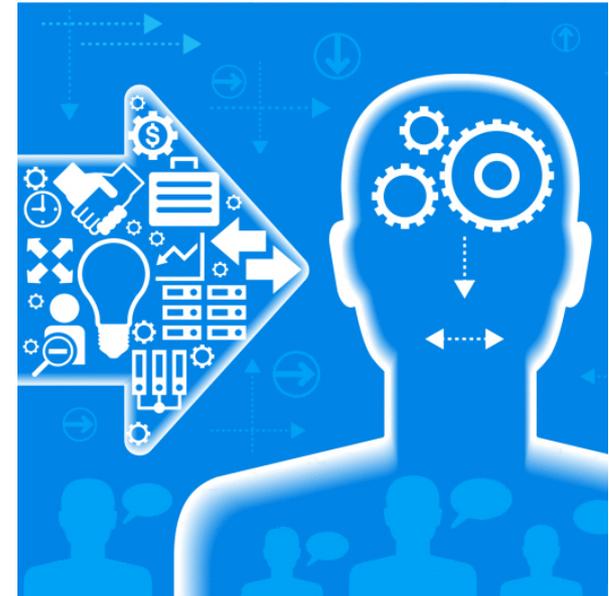


- Create User Stories with PwD use cases
  - Utilize PwD personas
    - <http://curbcut.net/accessibility/personas-of-persons-with-disabilities/>
    - <http://www.w3.org/WAI/redesign/personas.html>
  - Keep PwD goals and challenges in mind
- Put accessibility into each use case
  - If funding or features are cut, each feature has accessibility built in
  - Avoids problem of: we had to cut something, so we cut accessibility
- Make accessibility part of unit testing





- Pushing accessibility into design and toolkits
- One UI™ – Standard for design
  - Consistency
  - Experts agree on design
  - Accessibility experts input
- Pushed into toolkits
  - Dojo toolkit release
  - Accessible web templates
- Personas usage in One UI™
  - Account for usability testing
- Gives designer pieces to work with
- Tools support UI designers



*RESULT: Accessibility “baked in” to designs*



- Jackie – a Linux System Analyst
  - His responsibilities
  - His education and previous jobs
  - His interests outside of work
  - His coding skills and preferences
  - His work environment
  - Who he works with
  - The fictional firm he works for
  - Responsibilities
  - Goals
  - Pain points
  - Tools he uses
  - Even an image
  - ... and his disabilities
- You'd swear you knew him by the time you finish reading.





- Co-location principle is to put teams together in the same place to gain synergies in solving problems
- Break work into units and assign teams with developers, UI experts, Accessibility experts, testers and PwD users
- Team members have access to other team member skills
- Effective for repetitive cycles (see Case Study 3)
- Effective for estimating work and getting stories right (Case Study 2)
- Dedicated remote teams can work when physical co-location impossible





- Inaccessible product needed work estimates for adding accessibility
- True estimate would require much of the work to actually be done: full testing, defect creation, defects estimated
- Teaming to review problems with PwD, discuss solutions, broad estimates on solutions.
- PwD expert used existing system to the extent possible
- Sighted individuals watched, guided, described when PwD was lost
  - Set up pages
- Used desktop sharing software and phone conference to co-locate



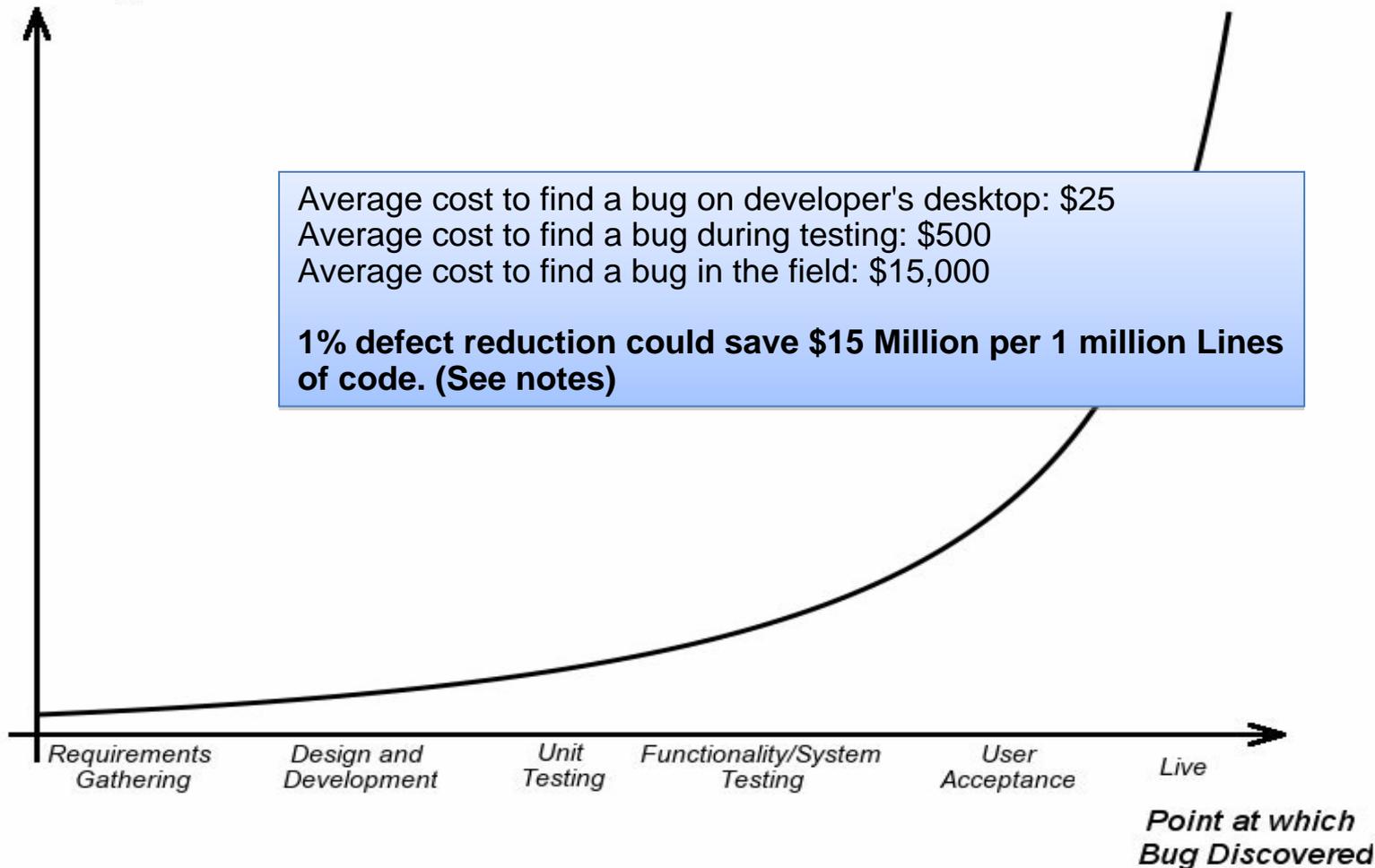
*RESULT: An effective review of major pain points and missing function provided enough understanding for a rough estimate.*

# Build accessibility in



Errors cost less to fix when they're found early.

*Cost of Fixing*



## Case Study 3: Early automated testing



- Tivoli Process Automation Engine
- Required to add WAI-ARIA
- Used static testing of Rational Policy Tester with each element update
- Most errors fixed by development team prior to release to test team
- Only 19 accessibility issues found in Accessibility Verification Test
  - Completed testing early
- By comparison other teams found upwards of 200 errors in AVT
- Developers happier – felt it was “effortless”
- Consistent with Continuous Integration (CI) practice of RAD

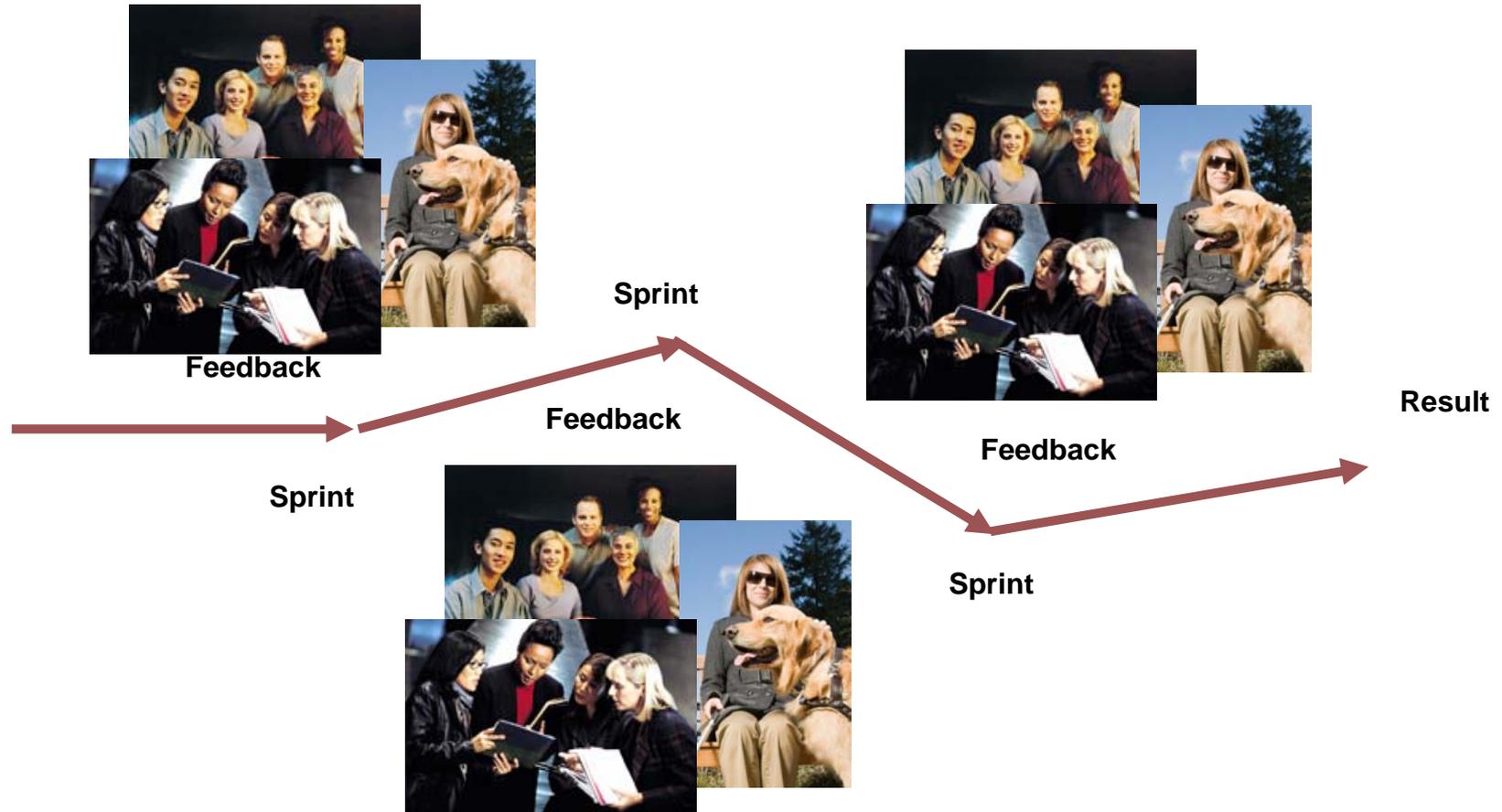


*RESULT: 16 products dependent on base-code became compliant*

*RESULT: 90% reduction in errors transferred to test team*



RAD principle is hands on feedback at each cycle to confirm results meet stakeholder needs. This allows course correction early in product development.

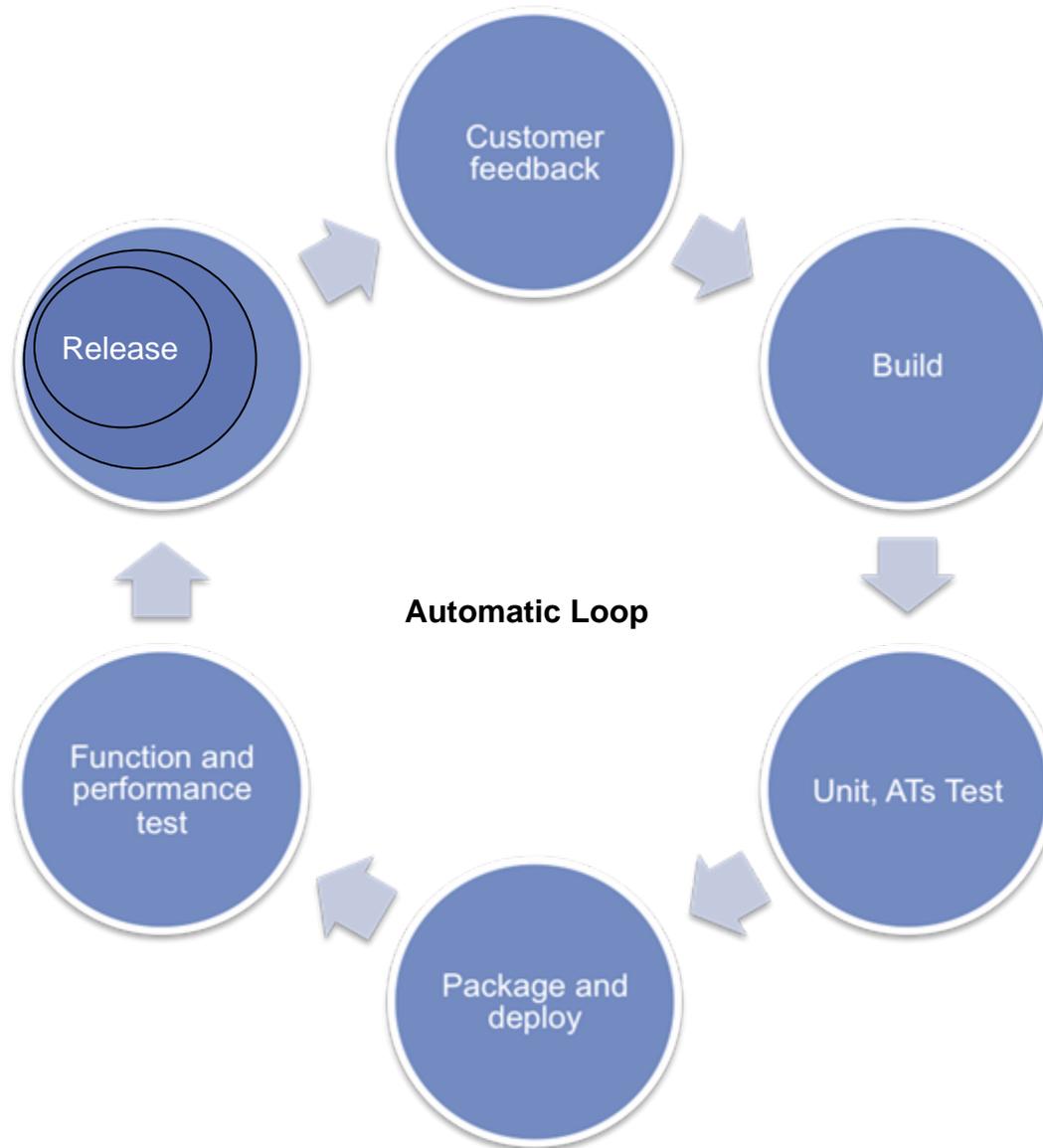


***Especially true of PwD stakeholders...***



- IBM Connections 3.0.1
- Needed more usable accessibility
- Weekly meetings with
  - Developers
  - ARIA expert
  - Accessibility expert
  - PwD
- Focus on specific features each review
- PwD attempted to use UI
  - Verbalized experience and expectations
- Others coached over rough areas
- Team fixed those problems during the week
- Reviewed the next week and went on to new areas
- Rapid cycle additions with frequent stakeholder feedback
- *RESULT: Best of breed usable accessible social collaboration system*
- This is repeatable: Replicated on one other product







- Characterized by many small frequent production improvements
  - Flickr (5-10 releases a day) small chunks pushed to production frequently.
  - Etsy (~20 releases a day)
  - Facebook (2x per day)
- Build the right thing
- Done means working code deployed
- Excellence, reliability, improvement at every step
- Automate as much as possible
  - Unit test
  - Other test
  - Build
  - Deployment
- Fast (often immediate) feedback
- Fix problems fast
- Ability to move quickly

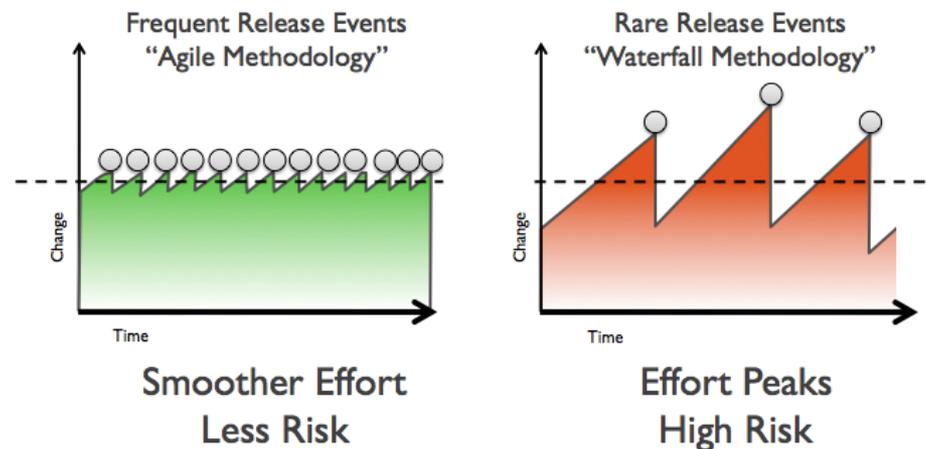


Photo Credit: Christopher Little / Wikipedia



*“...Continuous Delivery is not Continuous Integration. Continuous Delivery is being in the position to ship your product whenever you want, day or night...”*

- Neal Ford

*“... In 2011 software that cannot be automated, is broken. Get new software...”*

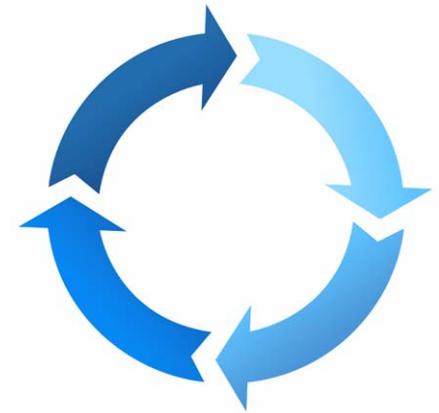
- Neal Ford

*“... Every time you do something for the third time, automate it. You’ll be doing it a million times...”*

- Neal Ford

Great article and source for the citations:

**“Another look at Continuous Delivery/Continuous Deployment“**,  
by Doug Rathbone. [http://bit.ly/continuous\\_delivery](http://bit.ly/continuous_delivery)





*Disclaimer: We haven't done this... but the same principles would apply.*

“Done is in the hands of the user.”

+ Many small deployments

➔ Integrate accessibility into every story, every “chunk”, every deployment

Excellence in everything

+ Automate as much as possible

➔ Integrate automated accessibility testing into process; utilize manual testing as needed

➔ Integrate accessibility into toolkit elements and development templates

Close to the user

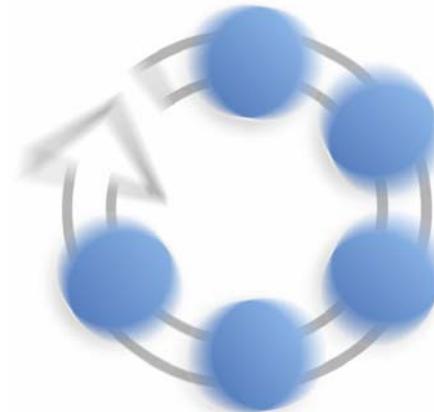
+ Quick feedback

➔ Seek input from PwD users

Test everything

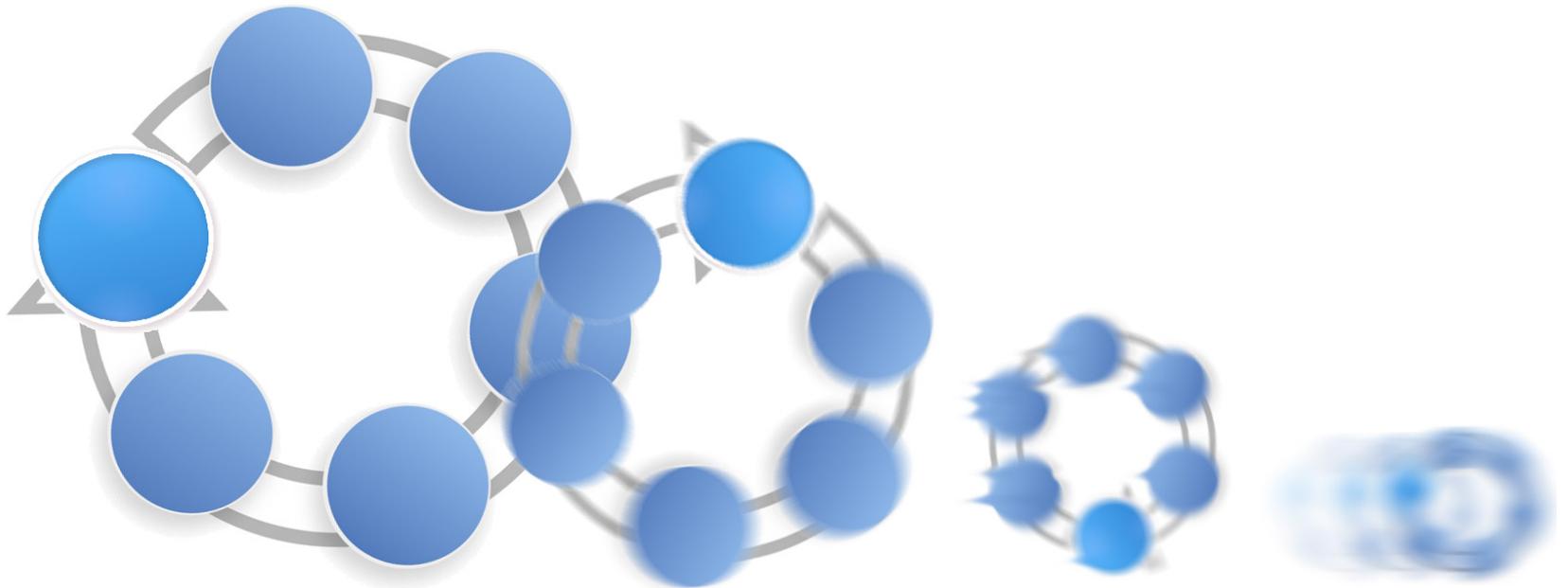
+ Eliminate errors of variableness

➔ Record use of tools, versions, etc. (e.g. JAW versions, browser versions)



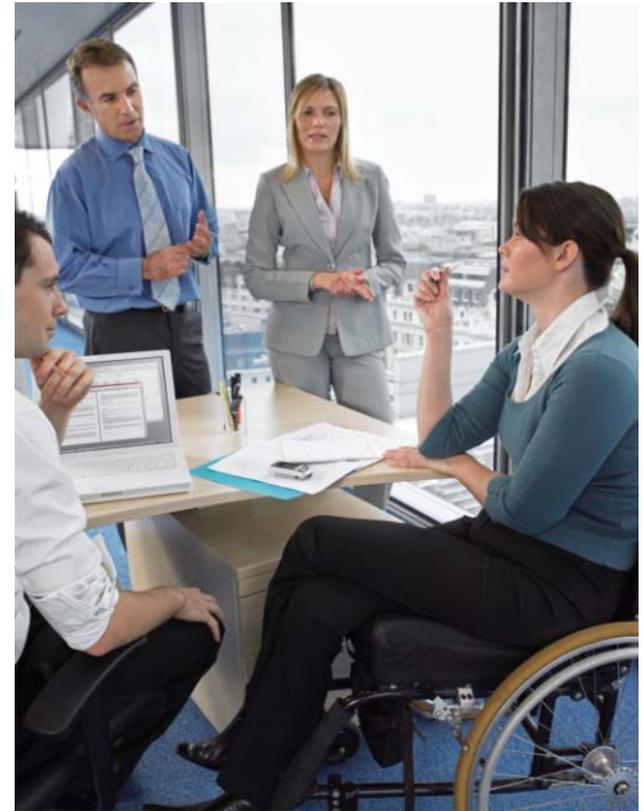


- Industry is shifting to faster deployment cycles
- Successful accessibility will come through
  - Integration into stories and chunks
  - Building into toolkit elements
  - Utilizing automation
  - Teaming to get combine skills and knowledge
  - Frequent stakeholder feedback
  - Looking for improvements in each step of the process
  - Accessibility advocate to encourage progress





- Determine the iterative process in use
- Determine current insertion points of accessibility
- Push accessibility earlier in the process / into more iterations
- Push for accessibility in requirements
  - Watch for platform / architecture decisions that don't consider accessibility
- Seek review of UI design.
- Push for accessibility in each work item so that if features are cut accessibility remains.
- Push for accessibility as part of early testing cycles.
- Push for SDK's to have accessibility built in.
- Utilize principles of colocation to bring the right teams together





thank you





The following trademarks are registered in the U.S. and other countries:

IBM, the IBM logo, ibm.com, OneUI, Rational, Rational Policy Tester, Rational Requirements Composer, Rational Quality Manager, Rational Team Concert are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries.

Find a list of IBM trademarks at “Copyright and trademark information” or [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml)