Managing Remediation of Web Accessibility Defects in a Large Enterprise

Presented by: Daniel Frank and Karl Groves

Argue for Budget by leveraging ROI

ROI = ((Risk amount – investment)/investment) \* 100

Where:

Risk amount = Expected Loss \* probability

Investment = Money spent on Accessibility

Target’s loss was 10 million (~7 mil in settlement + ~4 mil in legal fees) + loss to brand (known as “good will”)

What are the challenges?

Not all accessibility problems are equal

* Time
* Impact
  + Impact on users
  + Impact on business
* WCAG Level & Success Criteria is wholly inappropriate for determining priority
  + The WCAG 1.0 priority levels were based on user feedback, but didn’t take into account level of effort (i.e. political capital to change brand color schemes)

Challenges:

* Time
  + Often at a premium
  + Time spent on after-the-fact b ug repairs is time that is taken away from meeting other business needs
  + See, “Technical Debt”, Martin Fowler
* Impact
  + Budgets
  + Resources
  + System

Remediation approaches

* Simple prioritization
  + Focused solely on time and (simple) impact
  + How long will it take?
  + How bad is the problem?
* Advanced Prioritization
  + User Impact (across a variety of user populations, alt text only impacts a small population, but color alone as an indicator impacts blind **and** color blind)
  + Ease and Speed
  + Impact on Interface
  + Volume (how many instances of the issue?)
  + Location (part of a critical workflow, or an about us page?)
  + Secondary Benefit
  + Impact on interface and operation
  + Volume of repeat issues
    - How many times does the EXACT same issue occur?
    - How many times do (very) similar issues occur?
  + Location of issues
    - Traffic
    - Or criticality of location

Why Prioritize?

* Apply resources most effectively
* Minimize accessibility’s impact on business
* Motivate development staff
* Maximize positive impact for users
* Reduction of risk

Understanding Risk

* Ultimately, remediation of bugs is an effort at risk mitigation
* Risks of:
  + Poor quality (users having problems with system)
  + Loss of income
  + Ancillary losses
  + Administrative complaint (public sector)
  + Litigation

Developing a defect management program

* Have an accessibility testing program
  + Specifically supported configurations
  + Manage skills and expertise
  + Deploy testing tools, including AT
  + Document adequate levels of information
* Develop and test parameter definitions and weights
  + Identify application(s) for pilot
  + Clear parameter definitions are key to defect scoring
    - Discrete values between 1 and 4
    - Unambiguous and easily understood
    - Adjusted for clarity and consensus during white boarding process
    - Text of definitions incorporated into SOR during implementation

Develop scores for each of these between 1&4 that is relatively non-contentious and reach a clear understanding that is quick to do

* + Severity
    - Experience of users with disabilities
    - Multipliers for number of affected populations
  + Frequency and importance
    - Frequency of occurrence
    - Importance of business flow
  + Ease of repair
    - Impacts on design team, back and systems
    - Ease of code repair
  + Hold whiteboard scoring sessions
  + Adjust definitions on the fly
  + Evaluate white boarding results and adjust definitions and weights as necessary
    - Sample basic definitions
      1. Blocker (4): User is completely prevented from accomplishing a task. There is no work around
      2. High (3): User can perform task only through alternate path or workaround; accomplishing the task is likely only possible for power users.
      3. Medium (2): User can perform task but doing so will be frustrating or time consuming. User may require assistance from co-worker or support staff.
      4. Low (1): user experiences inconvenience or moderate frustration
    - Multipliers
      1. A “High” (3) impact defect for two or more user populations should be regarded as a blocker and given a severity score of ‘4’.
      2. Otherwise the severity is the max of
         * The count of affected populations with an upward bound of 4 (blocker)
         * All severities assigned for this defect
* Implement scoring process
  + Clean up defects and defect documentation standards
  + Modify defect SOR to record parameters and weights
  + Create weighing pipeline or process
  + Train responsible staff and publish documentation
  + Priority parameters used
    - Impact on presentation logic
      1. Requires substantial redesign of both visual and functional presentation design
      2. Requires substantial redesign of either functional or visual design, but not both
      3. Requires minor redesign of either visual elements or functional design, or both
      4. Resolution does not require changes to visual or functional design
    - Impact on business logic
      1. Remediation requires changes to back end data architecture, such as database or information data schemas, with downstream effects on back end code implementing business logic, etc.
      2. Resolution requires changes to back end code architecture but not back end data architecture
      3. Resolution requires minor changes to back end logic, but does not impact code architecture or schemas.
      4. Do impact on business logic
    - Ease and speed of repair
      1. Defect is intermittent and/or very difficult to diagnose or requires global changes to resolve
      2. Defect is consistent and straightforward to diagnose, but requires local changes to the design of underlying code to resolve
      3. Defect requires changes to the underlying code but not its design
      4. Defect requires correction of typos or minor parameter changes
    - Volume of defect
      1. Defect rarely occurs, less than one in ten sessions
      2. Defect occurs once in 2-10 sessions
      3. Defect occurs at least once in every user session
      4. Defect occurs multiple times in every user session
    - Location of defect
      1. Defect is located in an error flow of a validation-based or non-critical business flow
      2. Defect is located in the functional flow of a seldom used or non-critical business flow
      3. In error flow of a frequently used business flow
      4. Frequently used or critical business flow
* Develop ‘goodness’ metrics and defect repair policy
  + Identify denominator for ‘norming’ defect scores
  + Develop comparability metric
  + Set thresholds or other policies for fix vs. leave as-is
  + Measure and report regularly
  + Modify system of record
  + Create a weighing process
    - Things to think about when evaluating weights
      * The initial scores are relative. The absolute values don’t mean anything with norming.
      * Does there seem to be a reasonable spread of the scores? A good weighting scheme provides significant discrimination between high and low priority defects.
      * Do the scores make sense in relation to each other? In other words, do highly weighted defects seem like ones you would have chosen to fix first?
      * Are you seeing bands of values that could be useful in setting thresholds in a defect repair policy?
      * Are there odd results? If so, figure out why.

Good Defect Hygiene is critical

* A rigorous prioritization system demands more of your defect documentation
* One defect per record!
  + The same defect on a different page is a different defect
  + The same defect in different locations on the same page must be recorded separately
* Record the browser and AT versions with the defect
* When testing with screen readers, provide screenshots of transcript windows, and consider making video recordings
* Clean up your defect data before you try to score it.

Who has to participate in Scoring?

* Key skills required:
  + Assess usability impact on tasks completion by different user populations
  + Understand relation to business function of task flow where defect occurs
  + Estimate impact of repair on different aspect of the application
  + Predict the type of repair required to application code
* Type of participants
  + QA testers
  + Product managers
  + Application architects
  + Developers

Insights gaine4d from white boarding

* Progress is slow at first, but speeds up quickly
* A common language and methodology breaks log jams and increases insight
* So far the scores ‘feel’ right, and the few surprises make sense
* Faster is better, make decisions and move on