moz://a

Cache The World
Adventures in A11y Performance

2023-02-05 FOSDEM

Benjamin De Kosnik / Morgan Reschenberg
Firefox Engineering, Mozilla

Collaboration

- Mozilla's Accessibility Team
  - Morgan
  - Find the team on matrix at: #accessibility:mozilla.org
- Mozilla's Performance Team
  - Benjamin
  - Find the team on matrix at: #perf@mozilla.org

Agenda

1. Scope/Context
2. Intro to Rendering, A11y Architecture
3. Motivating Cache The World (CtW)
4. Measuring Performance
5. Future Work
6. Questions

1
Scope & Context

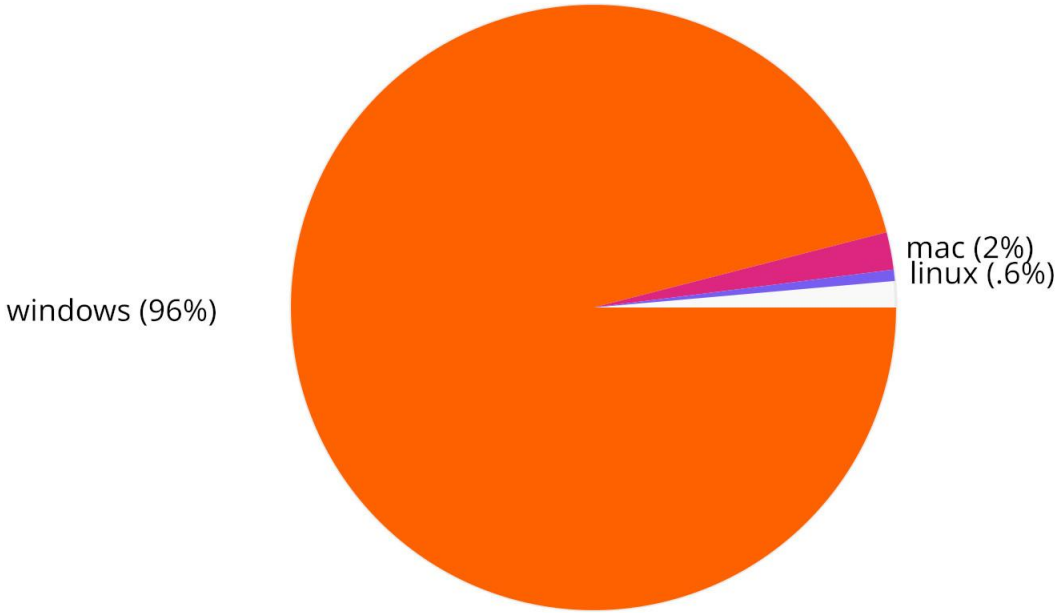
Goals

- Faster accessibility engine
- Performance Tests
  - what web content and test variations clearly show a11y use, non-use, implicit use, CTW-use?
- New Metrics / Public Dashboards
  - traditional W3C and PageLoad metrics, matching Profile Markers to Perf Stats
  - https://arewea11yfastyet.com/
- Documentation / Debug / Profiler Markers
  - a11y workflow, segmented to show up in web profiles taken with firefox profiler
- Collab Infrastructure
  - monthly meetings, chat

Scope

- YES: Screen Reader Performance
- NO: [Screen Magnification, Contrast Modes, On-Screen Keyboards, Subtitles] Performance

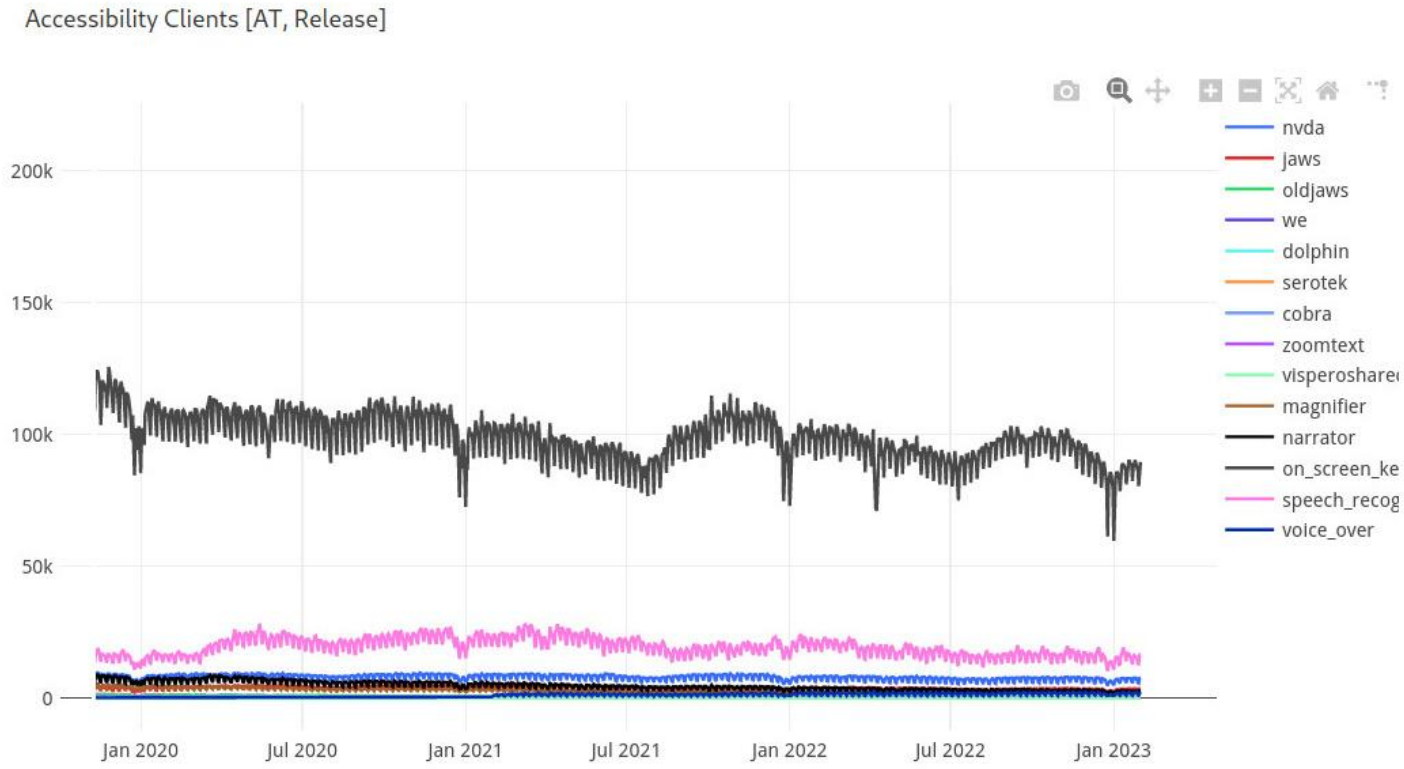
Context 1 OS Breakdown of all a11y-enabled page loads (Jan 2023)



Context 2 5.5 % all page loads accessibility enabled

5.8%	of windows
0.86%	of mac
2.74%	of linux

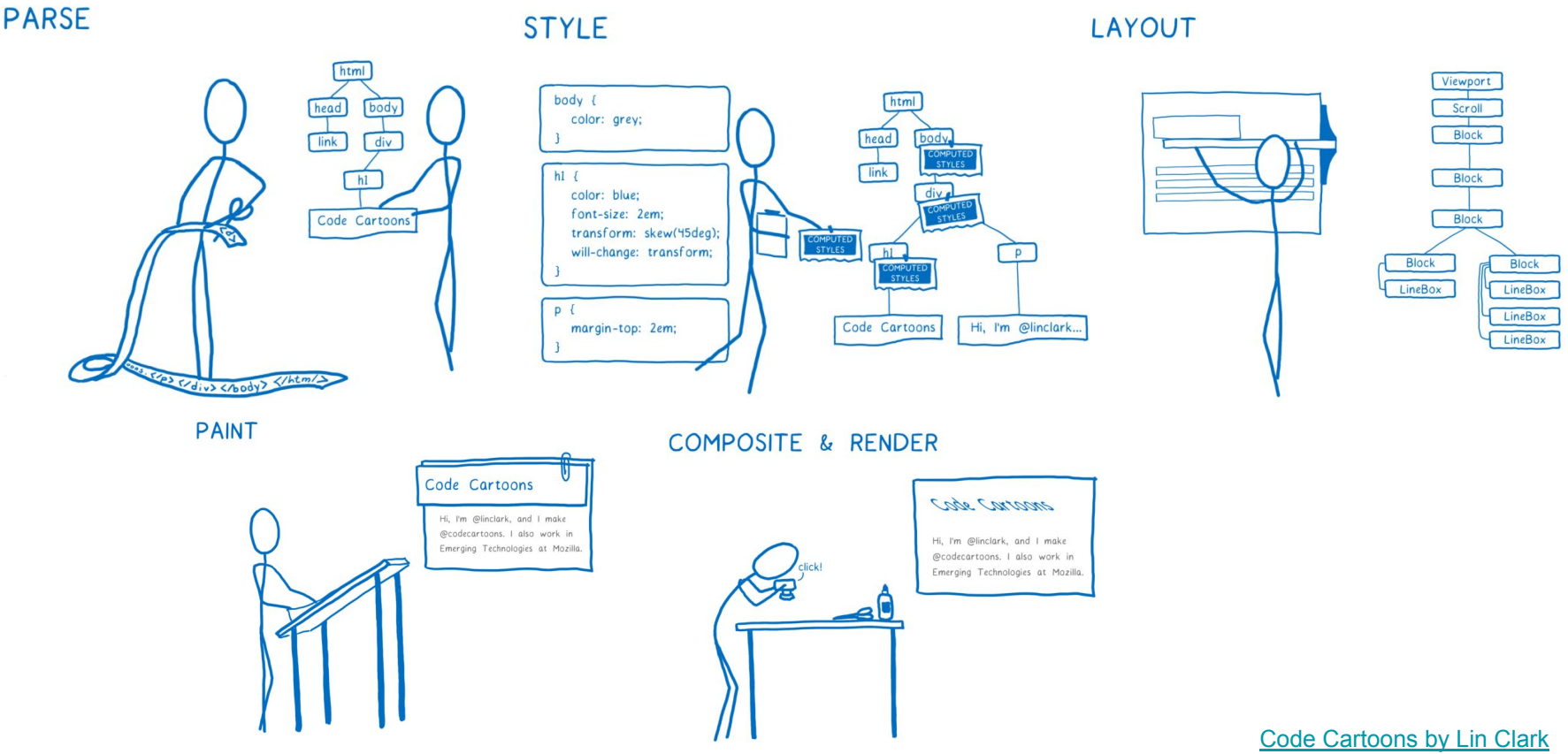
Context 3 Accessibility client breakdown



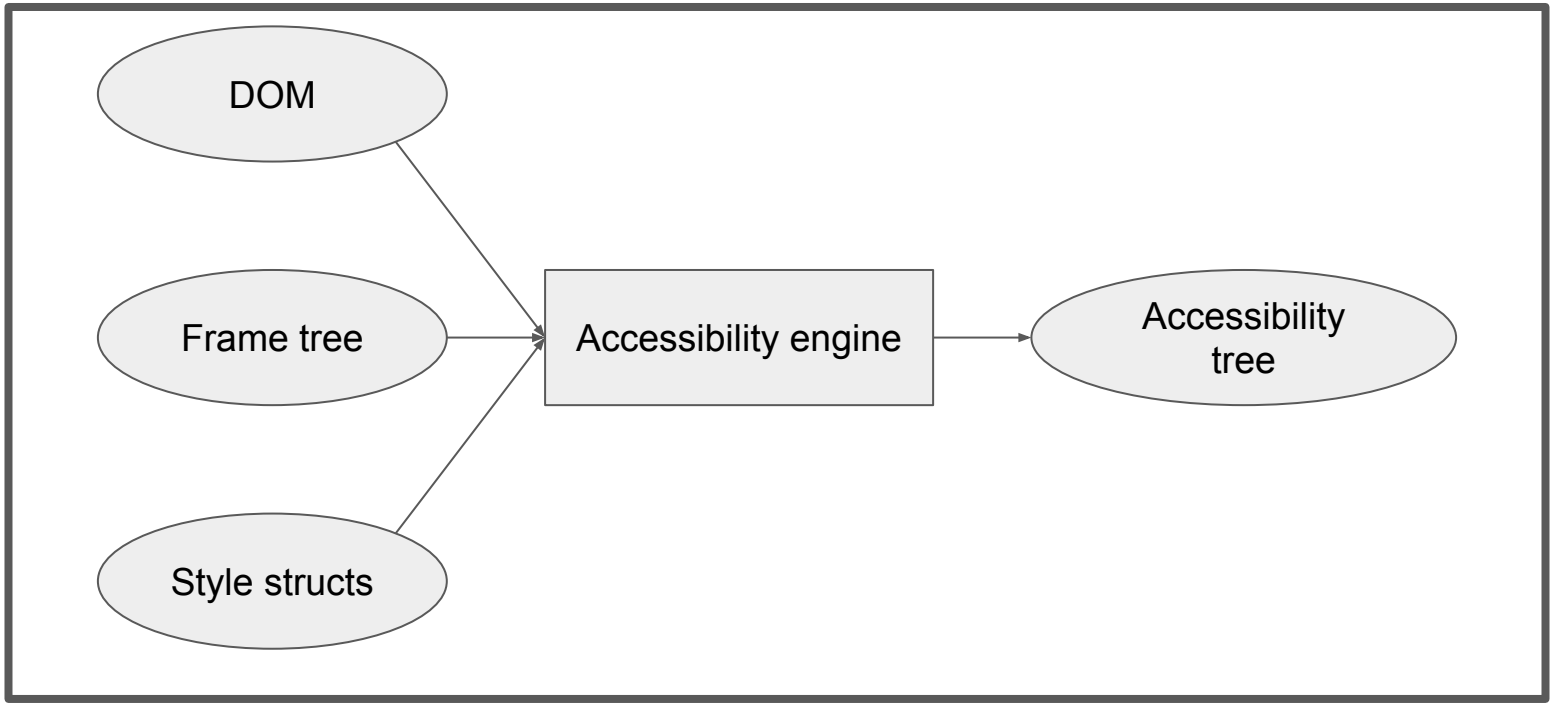
2

Intro to Rendering, A11y Architecture

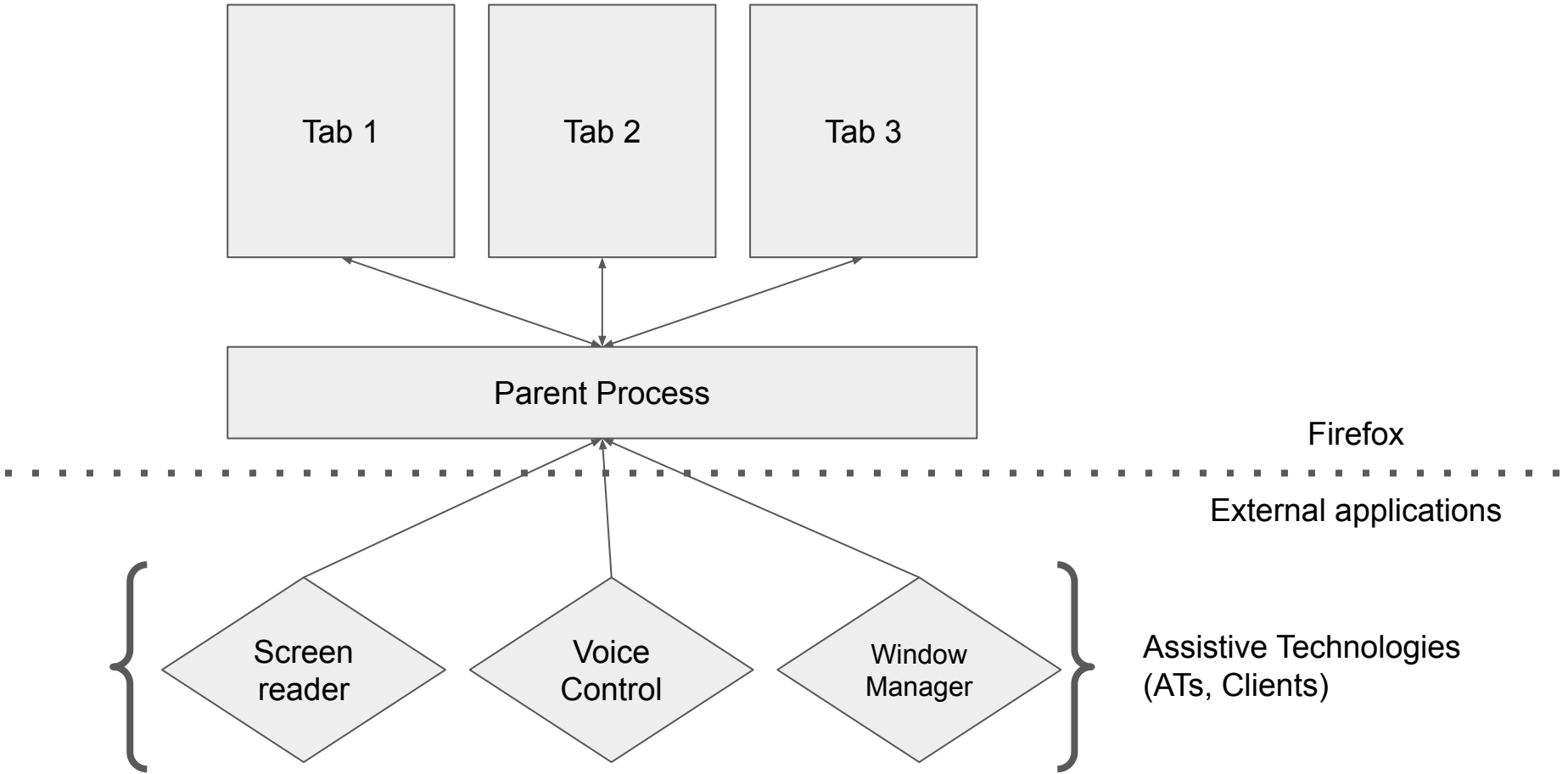
Browser Overview

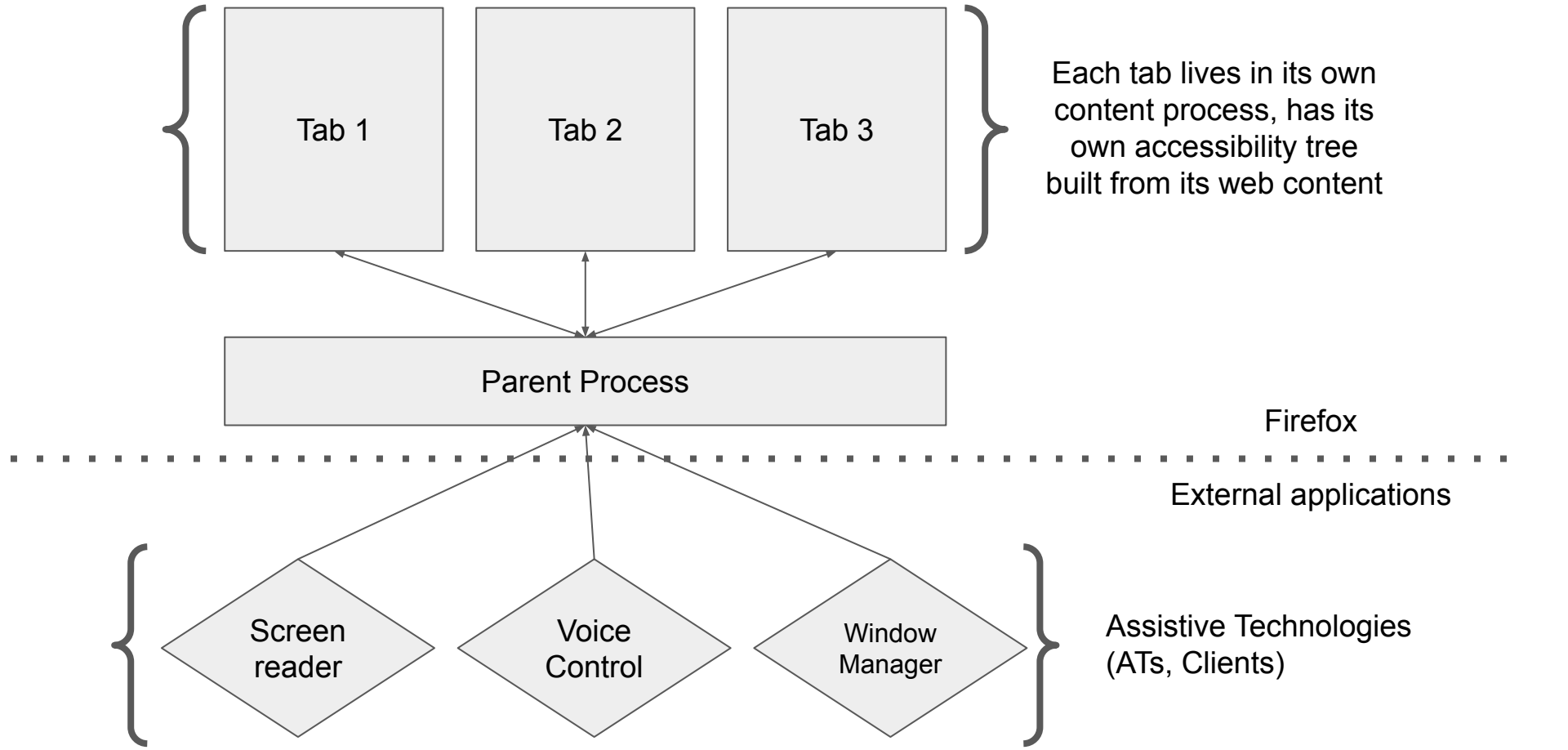


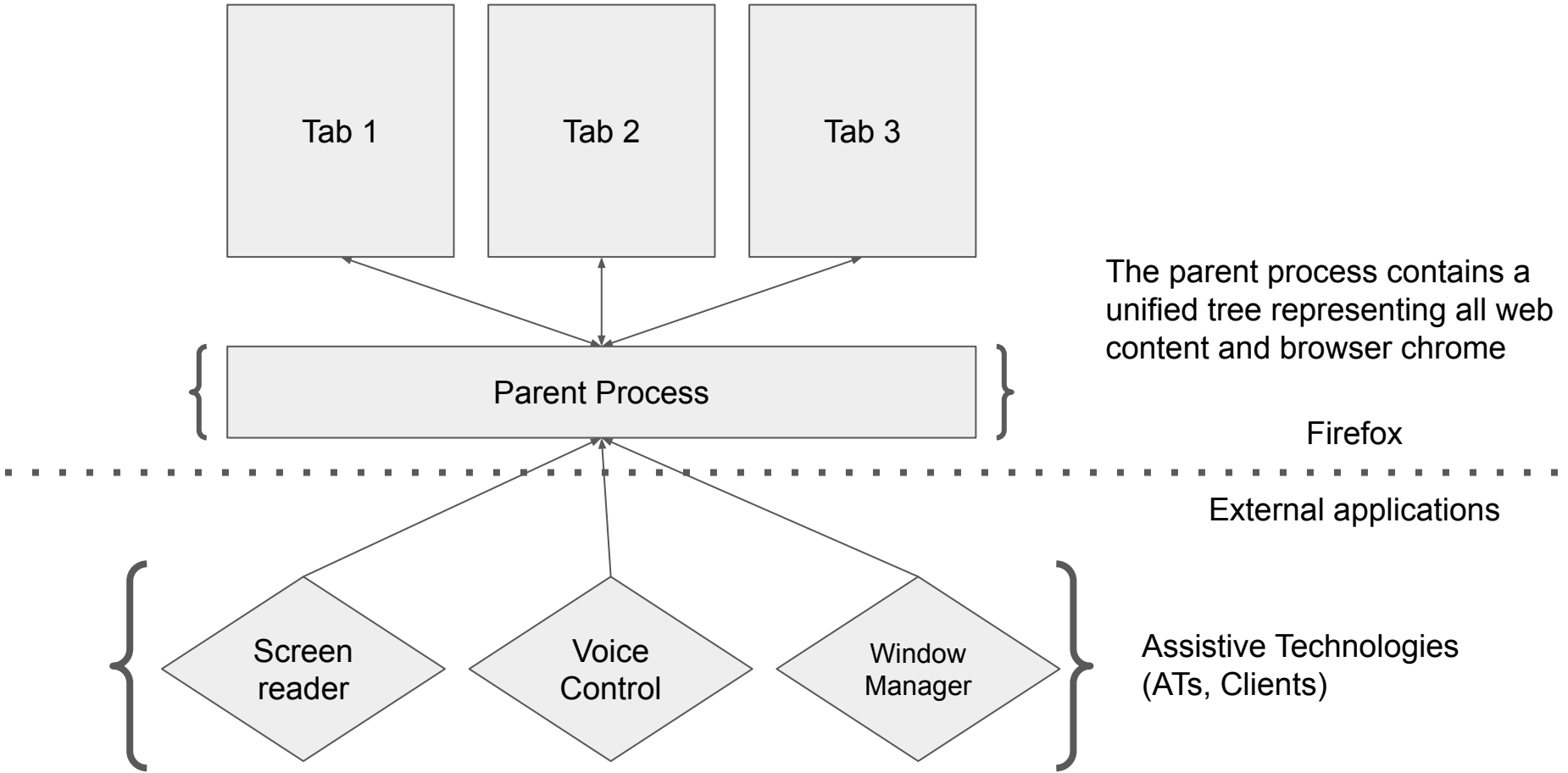
[Code Cartoons by Lin Clark]



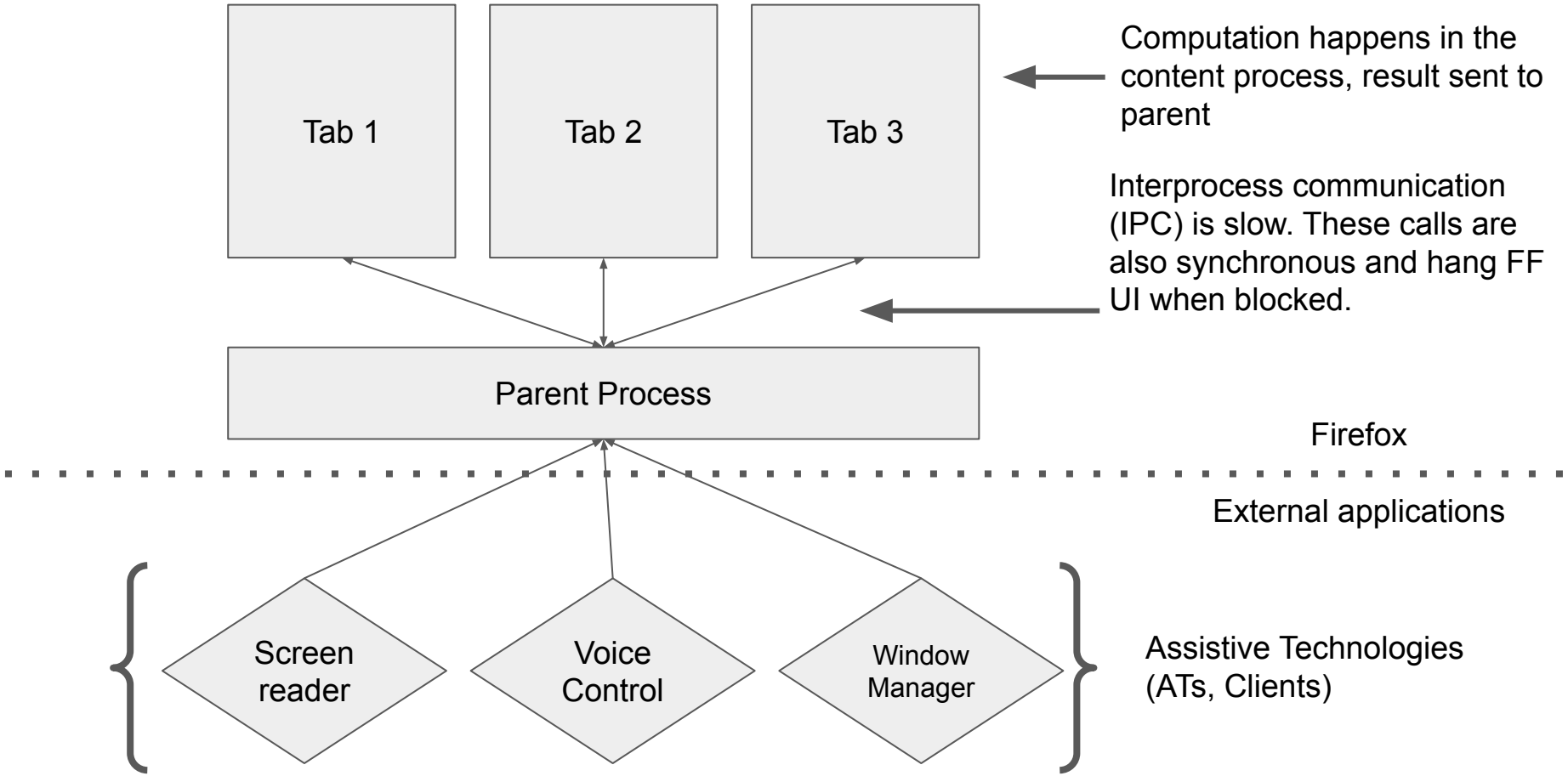
Content Process

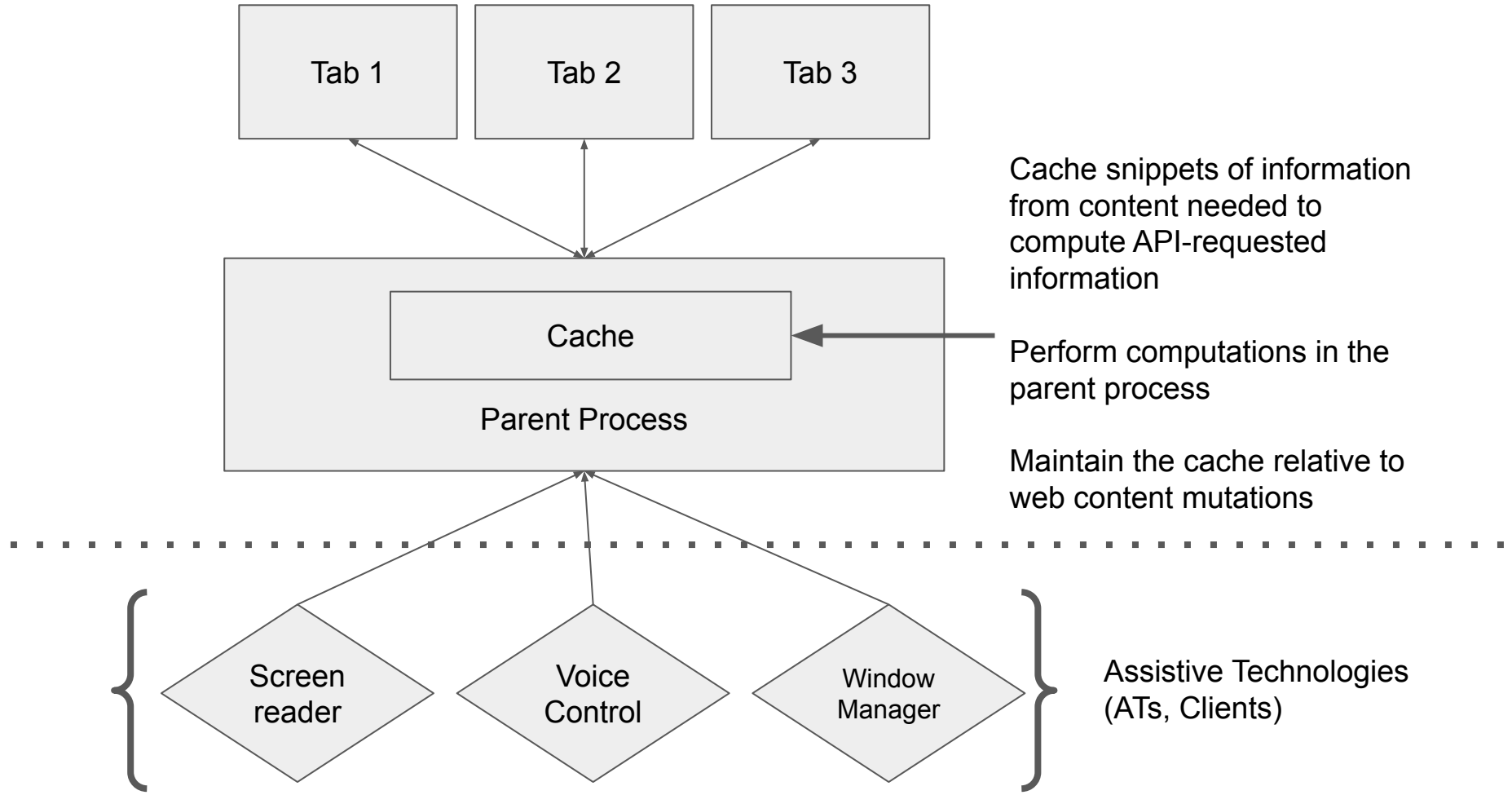

Tab 1
Tab 2
Tab 3
Parent Process
Firefox
External applications
Screen reader
Voice Control
Window Manager
Assistive Technologies
(ATs, Clients)


Tab 1
Tab 2
Tab 3
Each tab lives in its own content process, has its own accessibility tree built from its web content
Parent Process
Firefox
External applications
Screen reader
Voice Control
Window Manager
Assistive Technologies (ATs, Clients)


Tab 1
Tab 2
Tab 3
Parent Process
The parent process contains a unified tree representing all web content and browser chrome
Firefox
External applications
Screen reader
Voice Control
Window Manager
Assistive Technologies (ATs, Clients)

3
Motivating Cache The World (CtW)


Tab 1
Tab 2
Tab 3
Computation happens in the content process, result sent to parent
Interprocess communication (IPC) is slow. These calls are also synchronous and hang FF UI when blocked.
Parent Process
Firefox
External applications
Screen reader
Voice Control
Window Manager
Assistive Technologies (ATs, Clients)

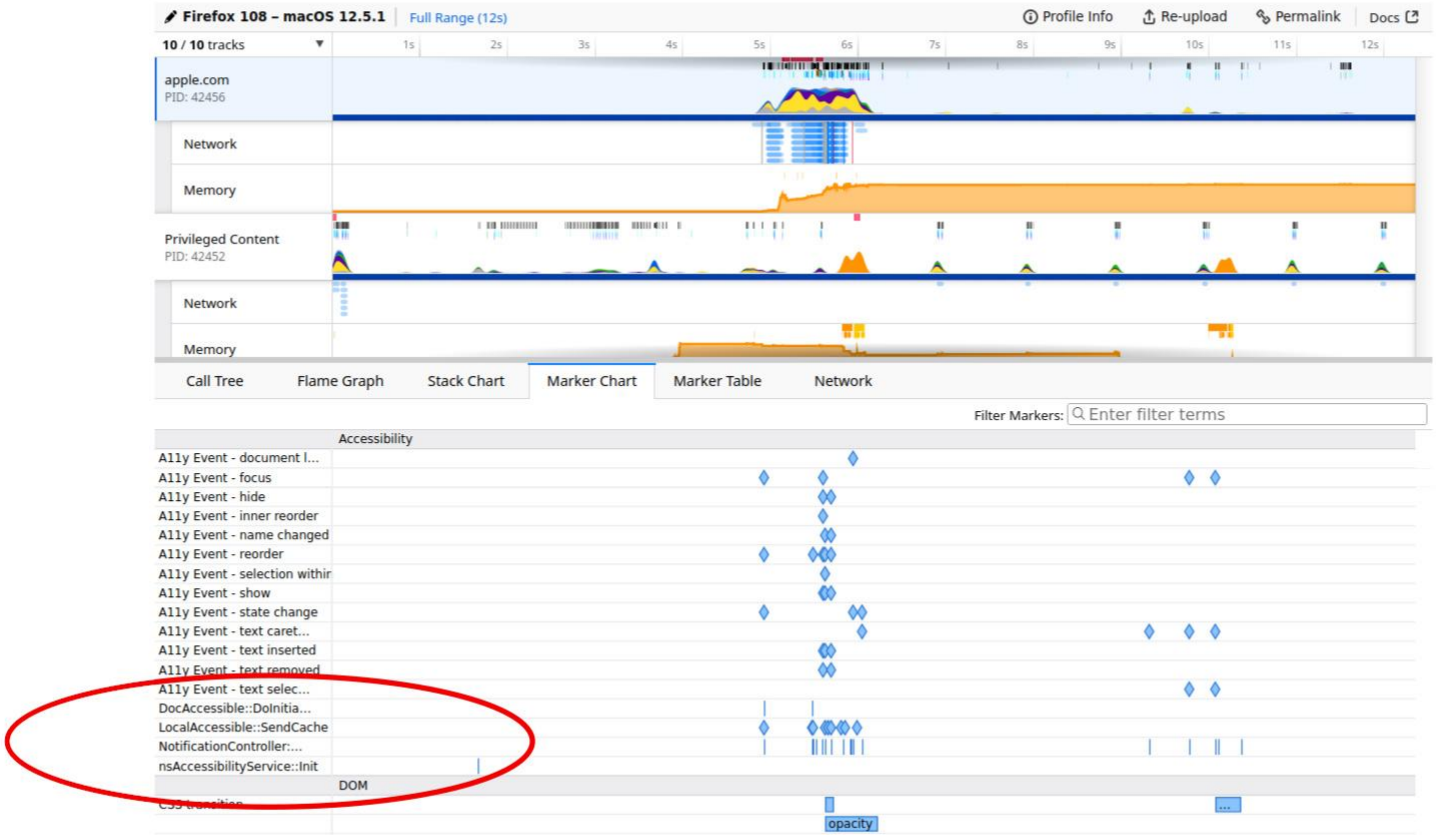

Tab 1
Tab 2
Tab 3
Cache
Parent Process
Cache snippets of information from content needed to compute API-requested information
Perform computations in the parent process
Maintain the cache relative to web content mutations
Screen reader
Voice Control
Window Manager
Assistive Technologies (ATs, Clients)

4
Measuring Performance

Profiler Markers

Markers for “important” code points for debugging A11y performance (via Michael Comella)

- NotificationController::WillRefresh
- nsAccessibilityService::Init
- DocAccessible::DoInitialUpdate
- LocalAccessible::SendCache
- ProcessQueuedCacheUpdate


Firefox 108 – macOS 12.5.1
Full Range (12s)
Profile Info
Re-upload
Permalink
Docs
10 / 10 tracks
1s
2s
3s
4s
5s
6s
7s
8s
9s
10s
11s
12s
apple.com
PID: 42456
Network
Memory
Privileged Content
PID: 42452
Network
Memory
Call Tree
Flame Graph
Stack Chart
Marker Chart
Marker Table
Network
Filter Markers:
Enter filter terms
Accessibility
A11y Event - document l...
A11y Event - focus
A11y Event - hide
A11y Event - inner reorder
A11y Event - name changed
A11y Event - reorder
A11y Event - selection withir
A11y Event - show
A11y Event - state change
A11y Event - text caret...
A11y Event - text inserted
A11y Event - text removed
A11y Event - text selec...
DocAccessible::DoInitia...
LocalAccessible::SendCache
NotificationController:...
nsAccessibilityService::Init
DOM
opacity

Testing Strategy, Web Content Selections

- Large OS/Screen Reader Variations
    - Mac, Windows, Linux, Android all have different OS strategies and devices
- Large Number of Test Variations (5)
    - baseline test, standard Firefox Nightly options
    - with accessibility turned on explicitly, cache off
    - with accessibility turned on explicitly, cache on
    - with implicit accessibility, cache on
    - with implicit accessibility, cache off
- Large Static Pages (initial cache push relative to document size)
    - wikipedia WW2
    - searchfox
    - whatwg html spec

Using Which Metrics?

- W3C Navigation Timing Page Load Metrics
    - Segments Browser Page Load into distinct phases, like the DNS phase of network, redirects, DOM parsing, content ready to display
    - Cross platform, useful to comparing to other browsers like Safari, Chrome
- Visual Metrics
    - Not measuring speech
- Perf Stats, implementation-defined micro measurements
    - Show promise as a way to measure A11y caching only

performance / loadEventStart Multi-Platform

Last 30 days

url_id searchfox-nscssframeconstructor-cpp

Legend

dots = cache off

dashes = cache on

performance / loadEventStart Multi-Platform

Last 30 days

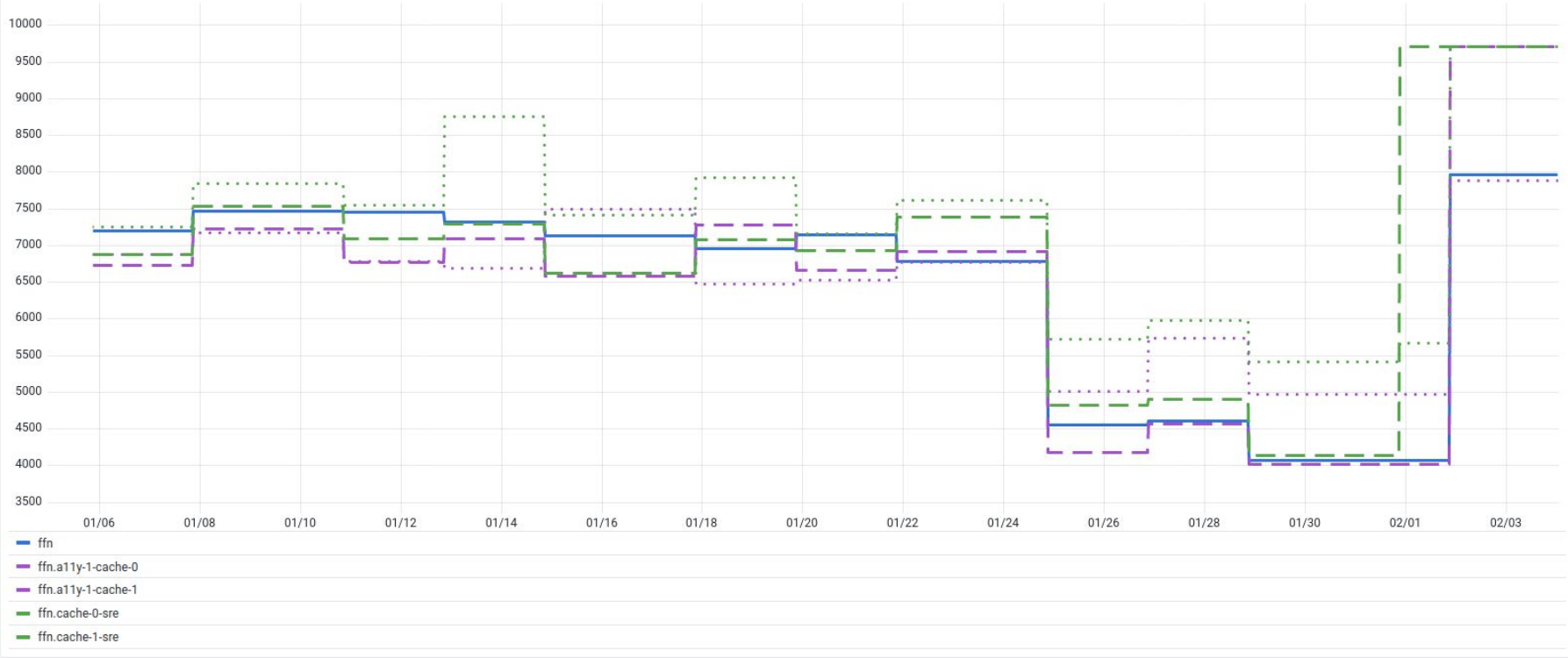
url_id imdb-tt0083943

Legend

dots = cache off

dashes = cache on

loadEventStart Multi-Platform



New (WIP) Metrics, Public Dashboards

- Align Profiler Markers to Performance Metrics
    - DocAccessible::DoInitialUpdate marker is beginning of cache creation
    - RemoteAccessibleBase::ApplyCache is end of cache creation
- Pay attention to Page Reload, Tab Switch Use Cases
    - Jamie Teh blog post on CtW
- Want Accessibility-First Metrics, not generic Page Load
- Public Dashboards for WIP
    - https://arewea11yfastyet.com

5
Future Work

A11y team roadmap

- Ship CtW to release (H1 2023)
  - Continue watching crash numbers, user feedback, etc.
- CtW optimisations based on performance metrics
  - Some “known” areas for improvement like caching granularity
- High contrast mode (HCM) data gathering and user-facing settings
- Datepicker and timepicker frontend accessibility improvements
- Ongoing collaborations with internal teams (design, frontend) to improve FF usability and accessibility

Performance team roadmap

- Ship CtW to release (H1 2023)
  - reproduce windows ad-hoc testing results
  - windows, android, mac testing in CI
  - tune markers, add more for end of speech range and a11y tree node counts
  - tune markers, add more for end of speech range and a11y tree node counts
  - Perf Stats to Telemetry page load event
- CtW optimisations based on performance metrics
- Ongoing collaborations with internal teams (perftools, etl, devops) to improve testing and data tools

6

Questions?

Thank You

Morgan Reschenberg (morgan@mozilla.com, @MorgynRae)
Benjamin De Kosnik (bdekoz@mozilla.com, @bdekoz@indieweb.social)

Resources and Notes

- Jamie Teh blog post on CtW
- Morgan's CSS Style Debug video
- bdekoz and color and contrast links