

# What do Researchers Need when Implementing Novel Interaction Techniques?

Thibault Raffailac, Stéphane Huot



22 June 2022

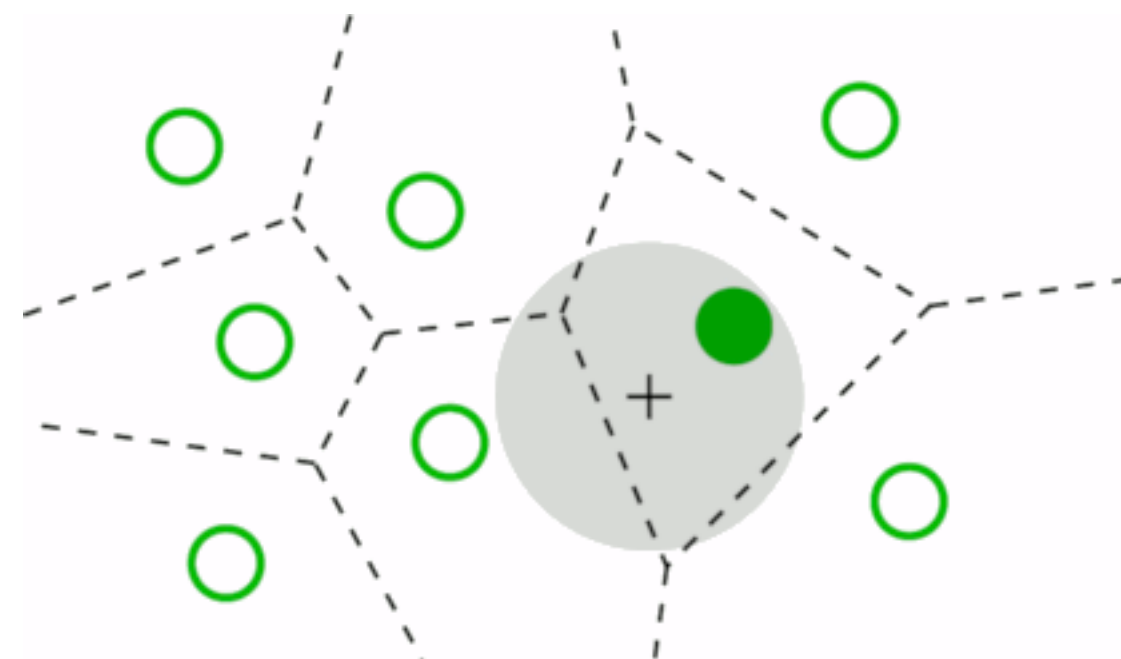
Who has ever programmed an Interaction Technique?

Who had trouble/frustration while doing so?

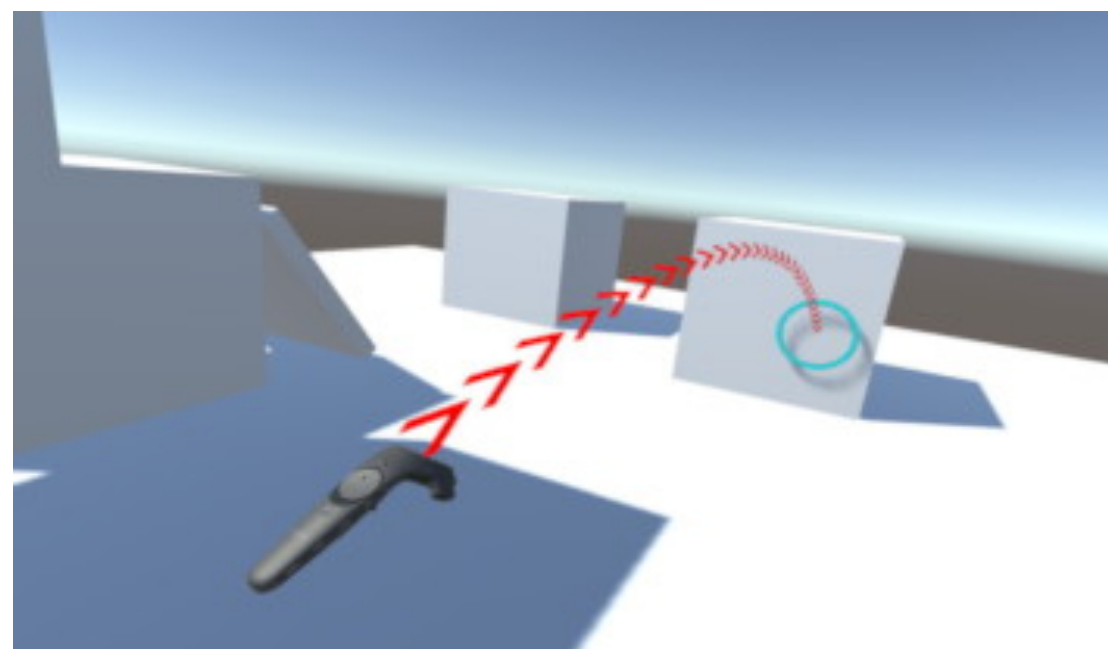
# Introduction

## Motivation

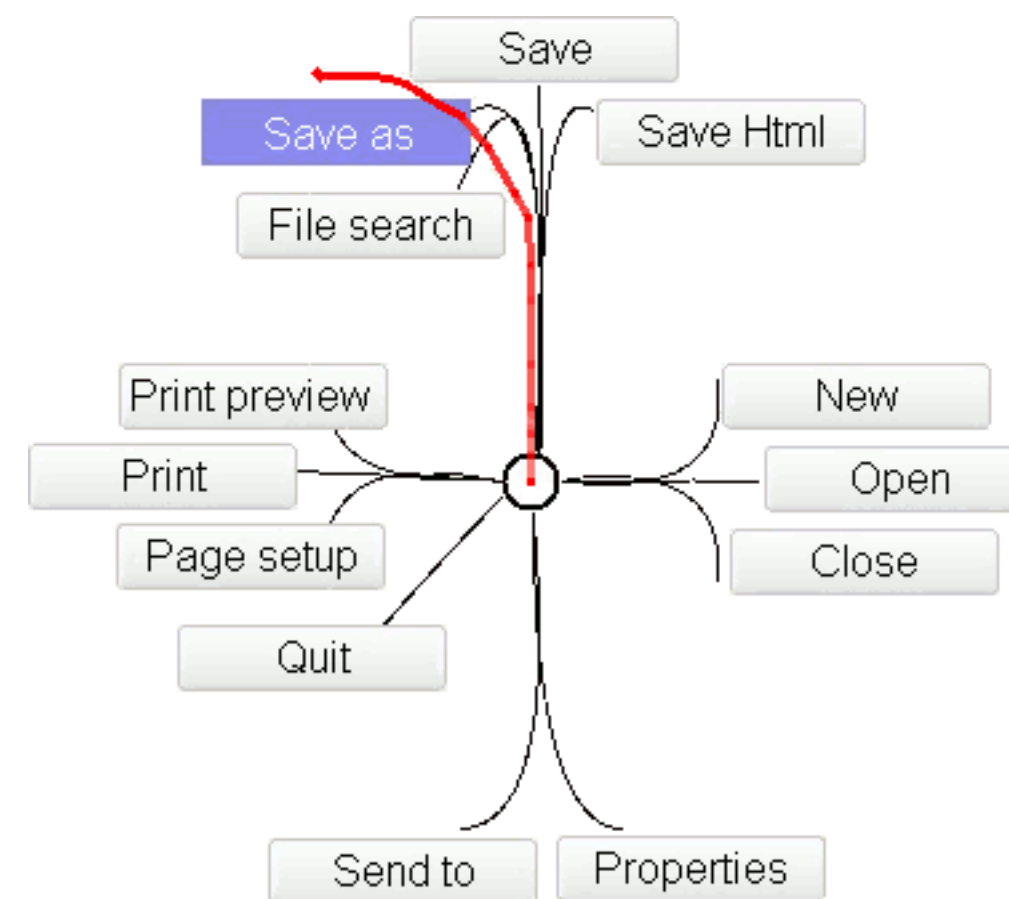
Frustration of colleagues when programming *interaction techniques* for research



Bubble Cursor (Grossman & Balakrishnan)



Unity VR Arc Teleporter



Flower menu (Bailly et al.)



ExposeHK (Malacria et al.)



Photoshop Lasso selection

# Introduction

## Problem

They may use:

- an interaction framework (Qt, HTML/JS, Swing)
- a research toolkit (D3, Amulet)

Frameworks are popular but:

- input data is hard to obtain
- insufficient granularity of reuse
- unchangeable behaviors
- lagging support for new devices

Consequences:

- limited adoption of innovative interactions (trackpad, gestures, eye tracking)
- recurrent publications of tricks to circumvent limitations (Prefab, Scotty)
- active research on toolkits/architectures as alternatives to frameworks

# Introduction

## Plan & Research questions

- Interviews & Survey

*What do researchers do when prototyping new interaction techniques?*

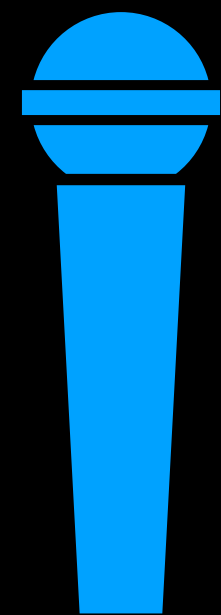
- Design recommendations

*How can we design or adapt existing frameworks and toolkits to support them?*

What do researchers do when prototyping new interaction techniques?

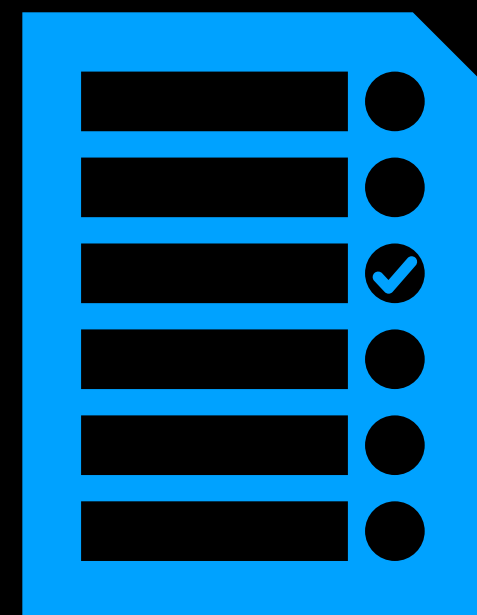
# Interviews & Survey

## Methods & Analyses



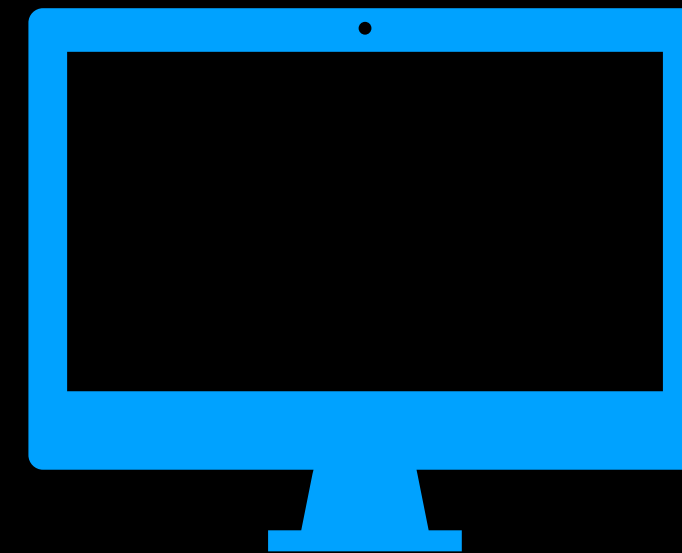
9 interviews  
Local researchers  
Semi-structured  
Problems with past projects

Thematic analysis



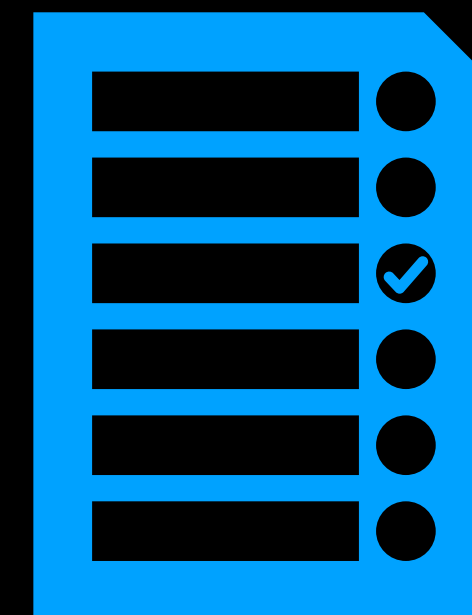
3 tables, 48 themes:

- problems
- utilities
- strategies



32 survey participants  
CHI community  
2/3 advanced or experts  
Rating predefined items

Quantitative analysis

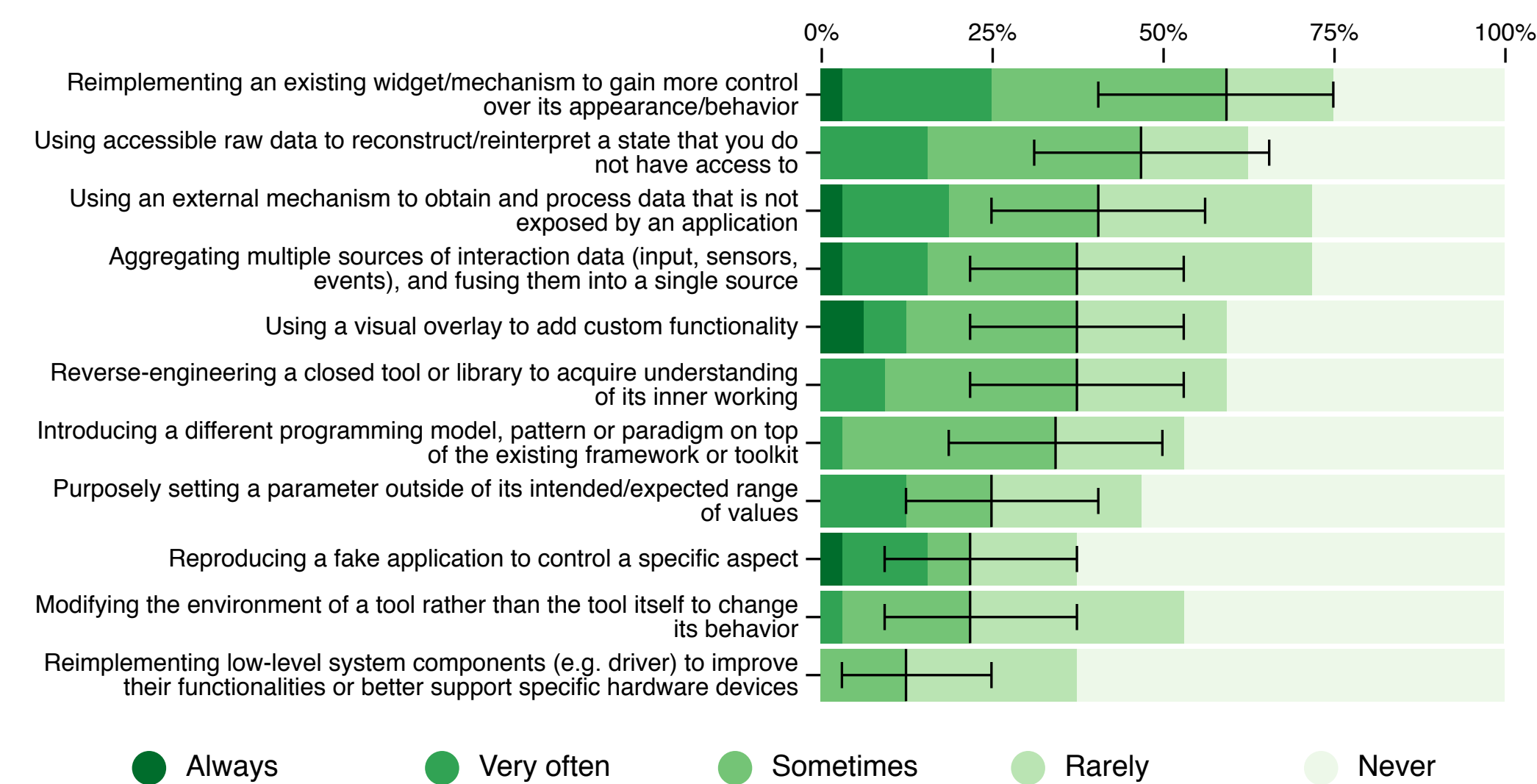
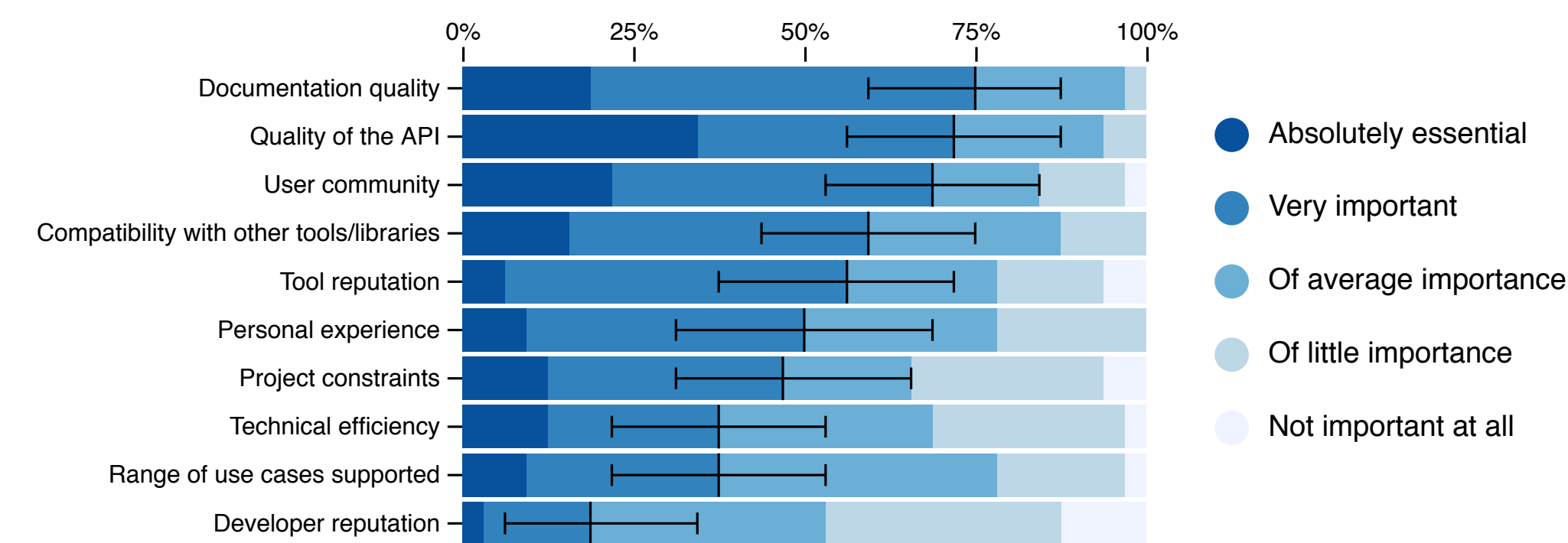
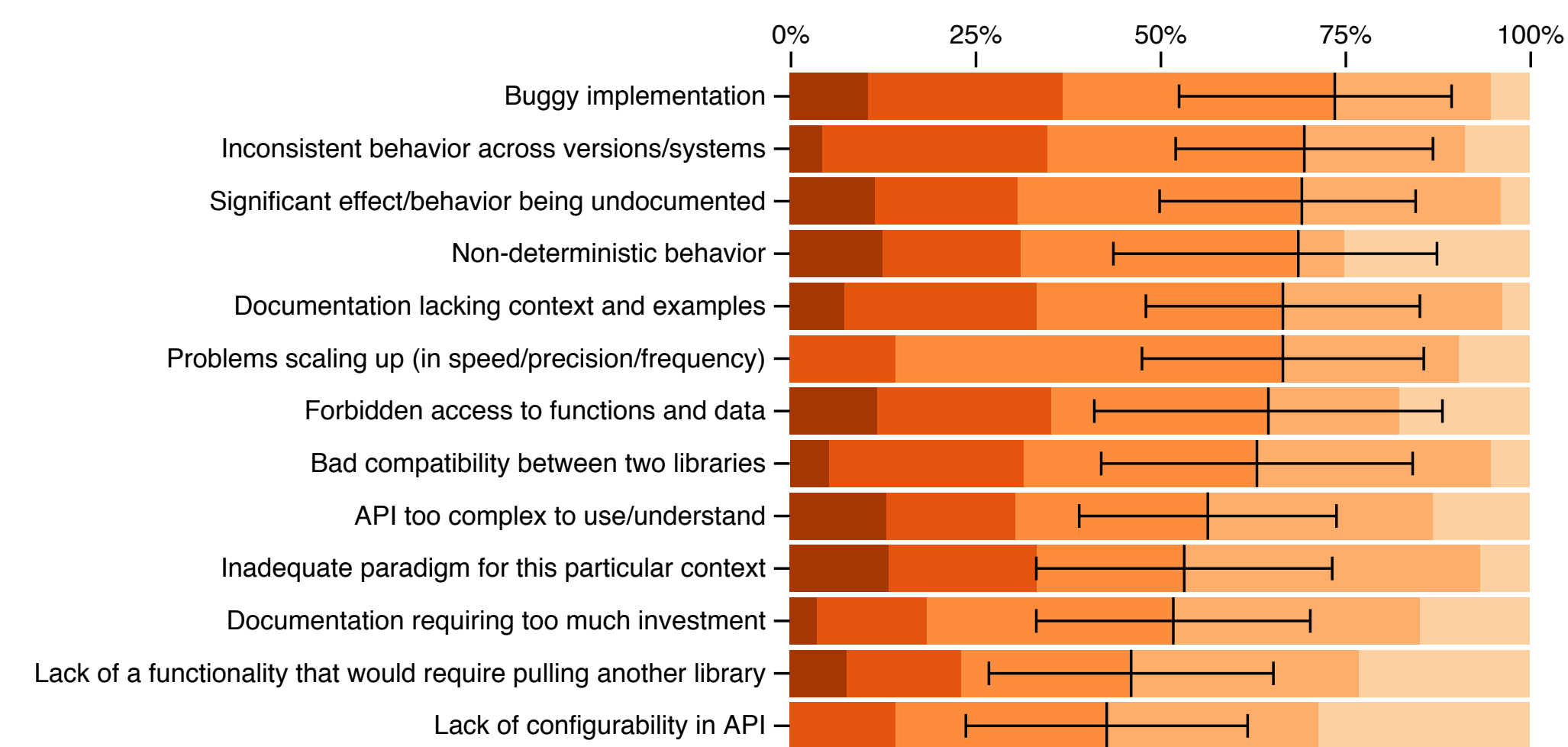
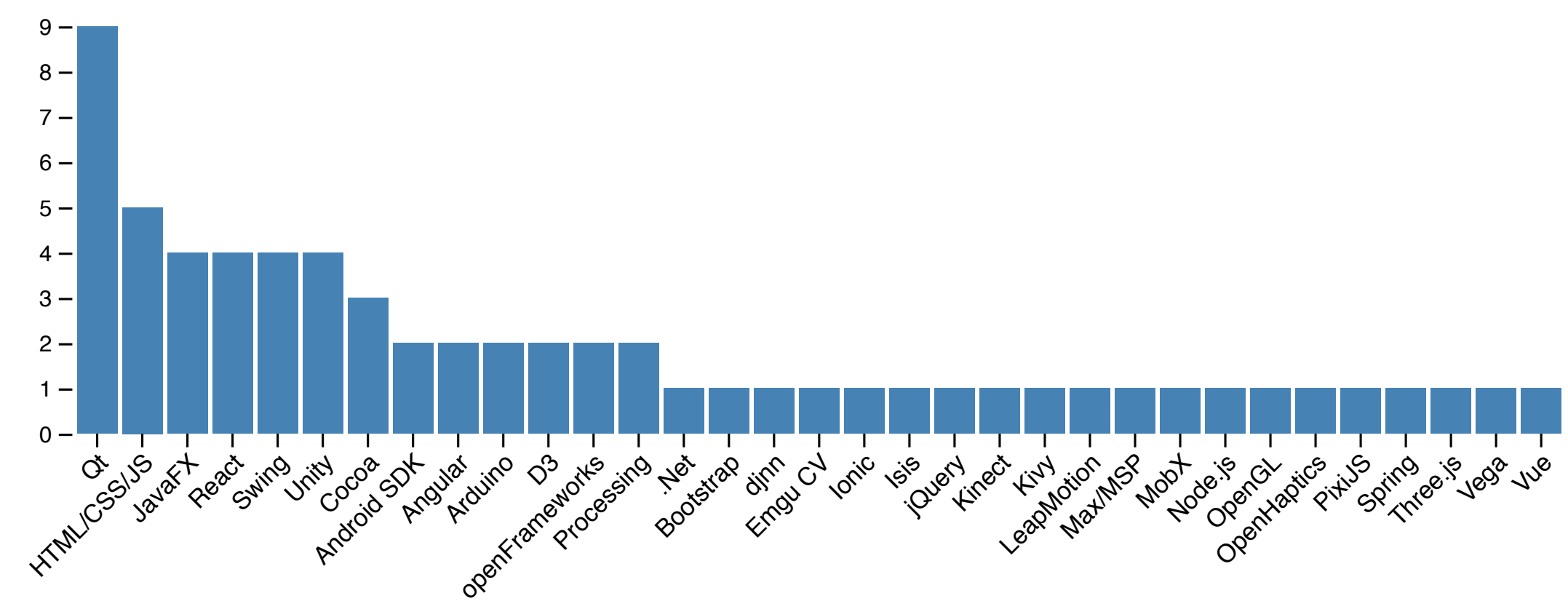


3 rankings:

- criteria of choice (R1)
- severity of problems (R2)
- frequency of strategies (R3)



# Interviews & Survey Results

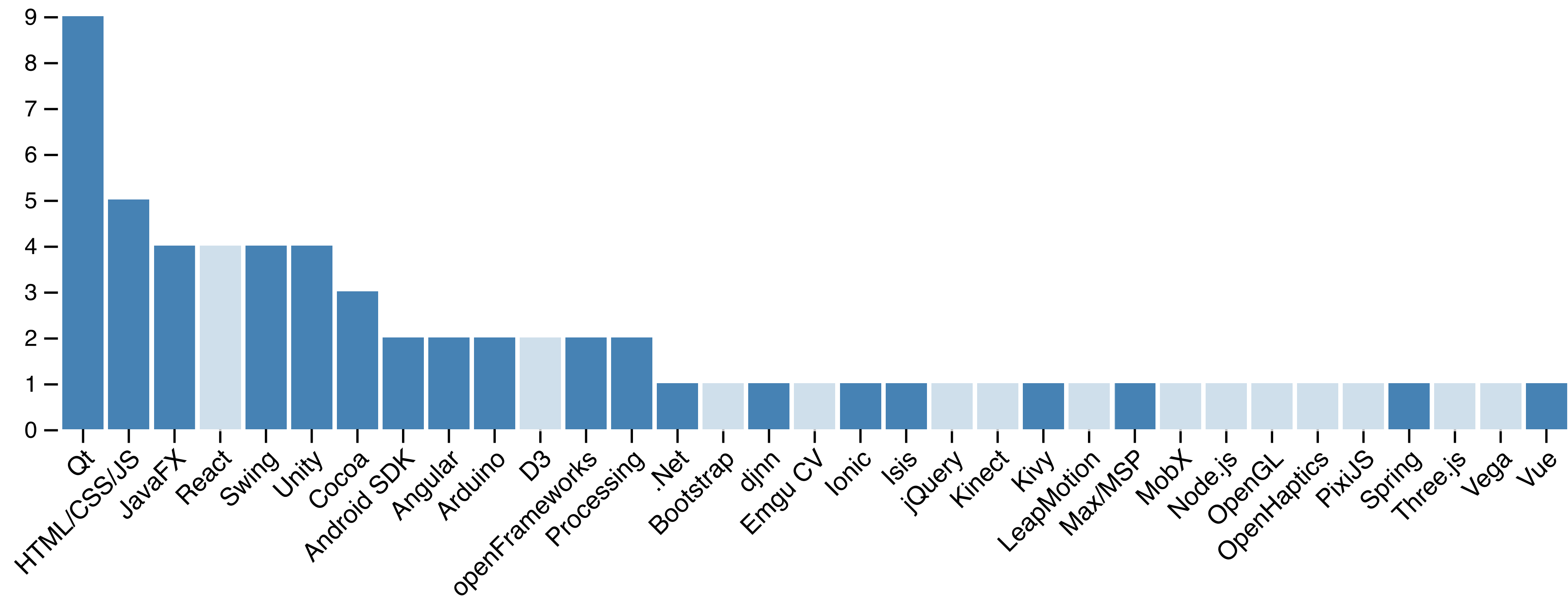




# Interviews & Survey

## Observation I

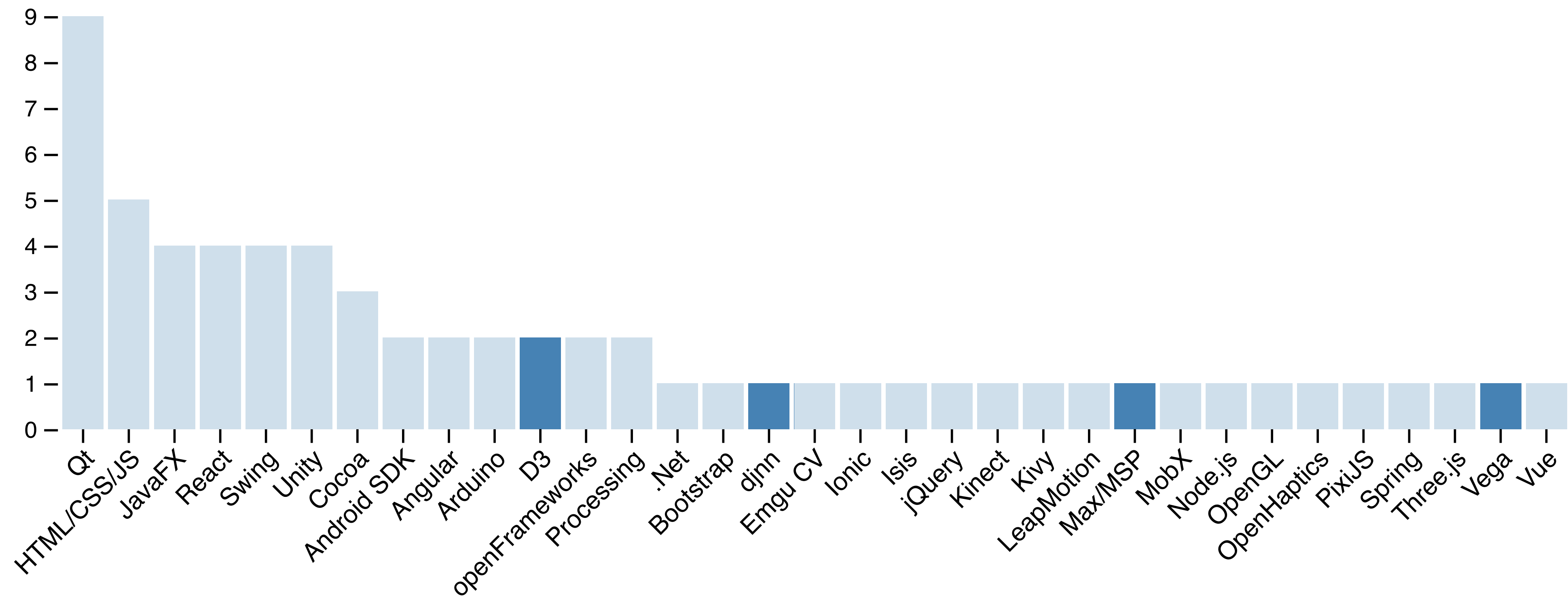
*Researchers prioritize well established interaction frameworks over research toolkits*



# Interviews & Survey

## Observation I

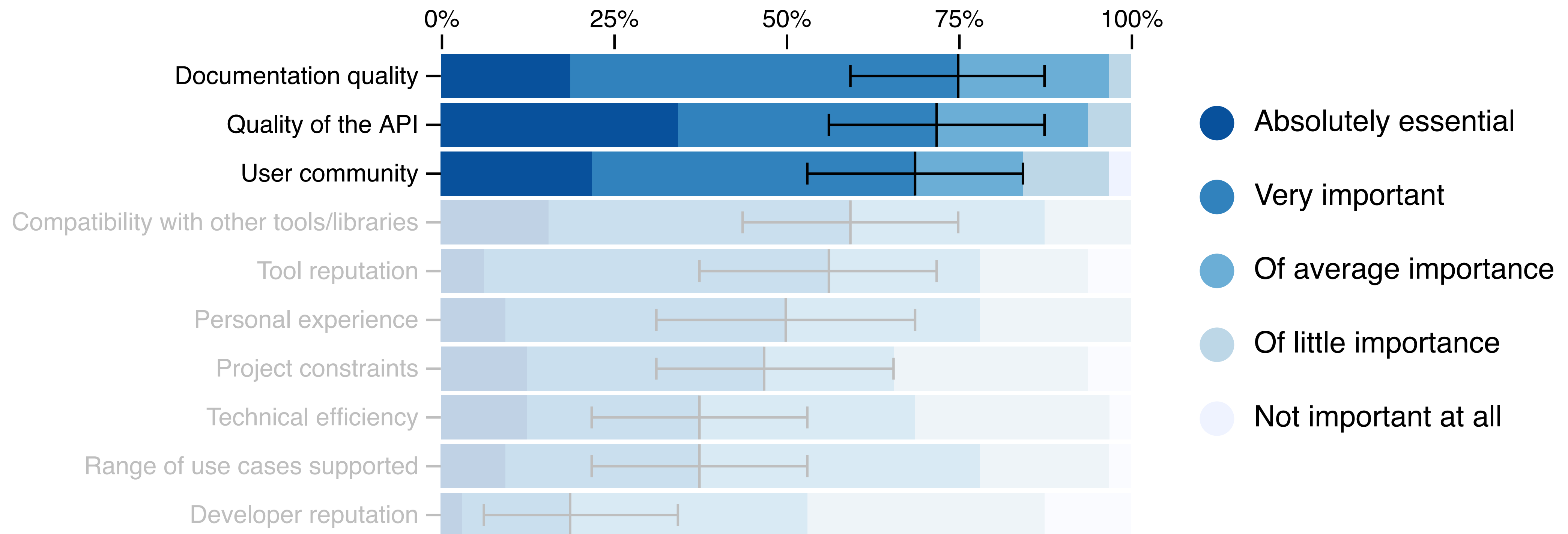
*Researchers prioritize well established interaction frameworks over research toolkits*



# Interviews & Survey

## Observation 2

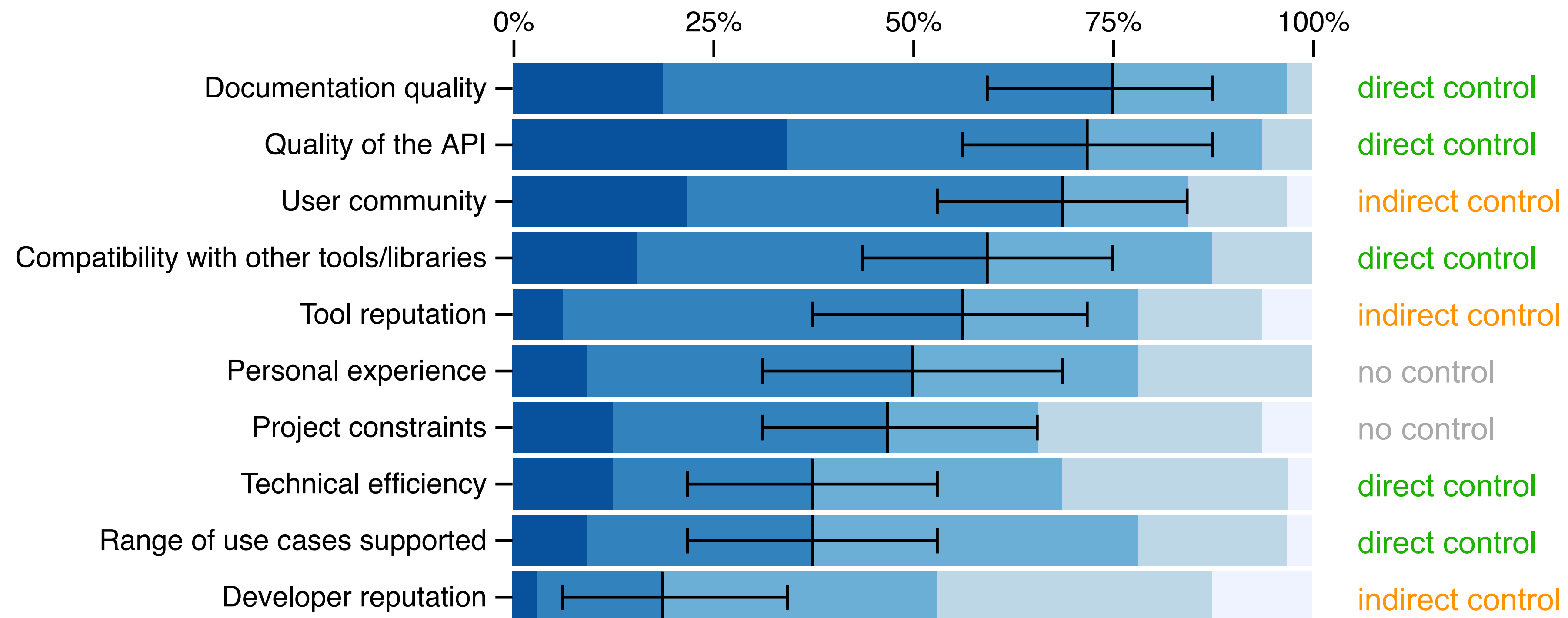
*The choice of a library is mostly based on its ease of use, and is directly controlled by its authors*



# Interviews & Survey

## Observation 2

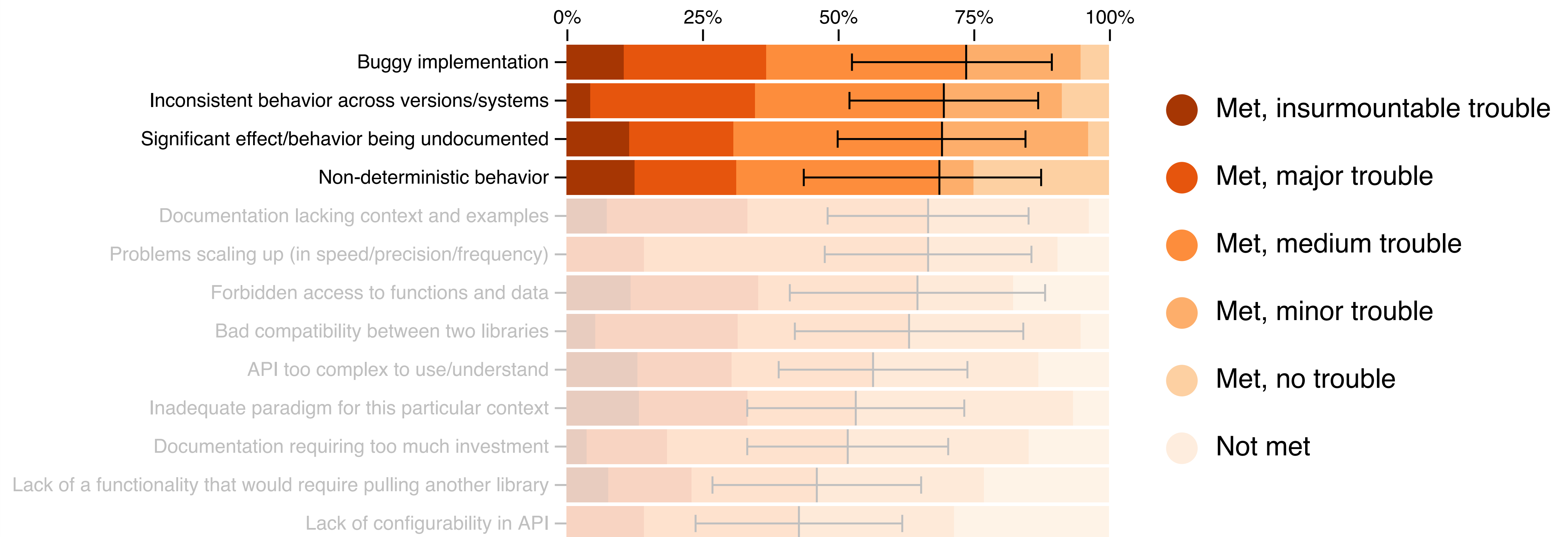
*The choice of a library is mostly based on its ease of use, and is directly controlled by its authors*



# Interviews & Survey

## Observation 3

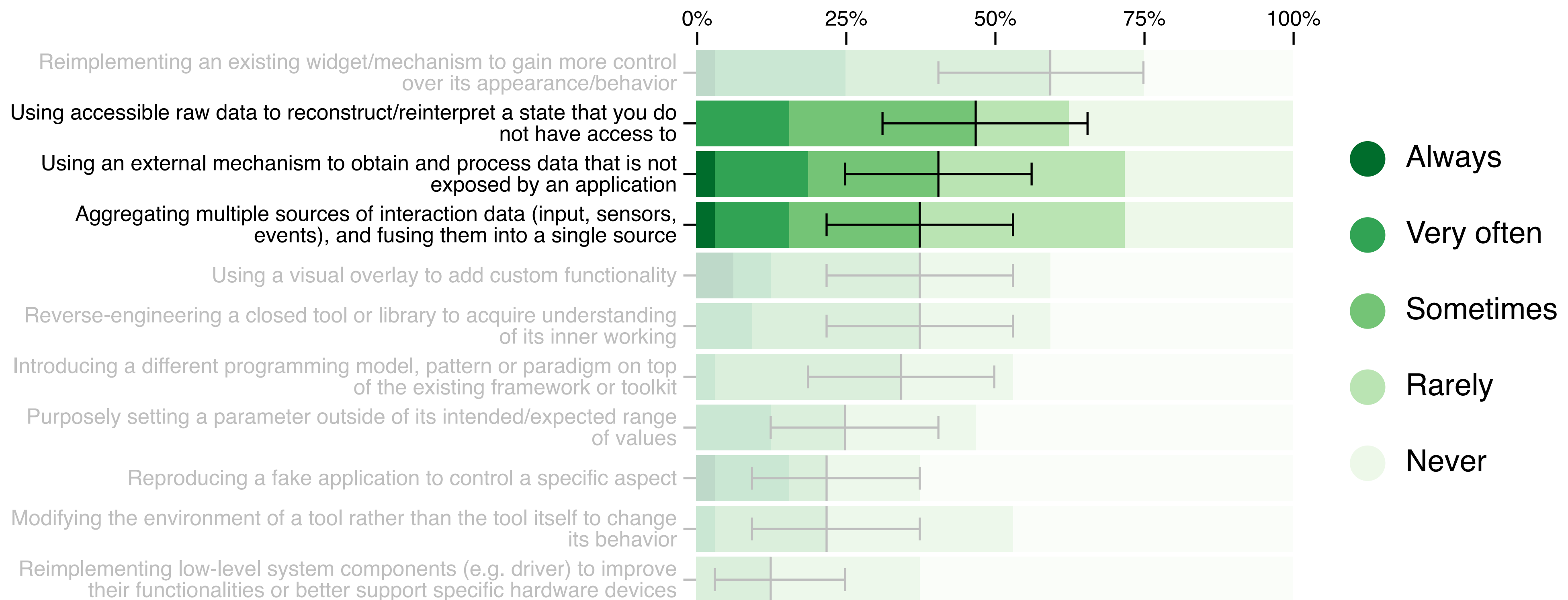
*Unpredictability is the most critical problem experienced by researchers with interaction libraries*



# Interviews & Survey

## Observation 4

*Strategies for gathering and processing interaction data are among the most frequent for our participants*

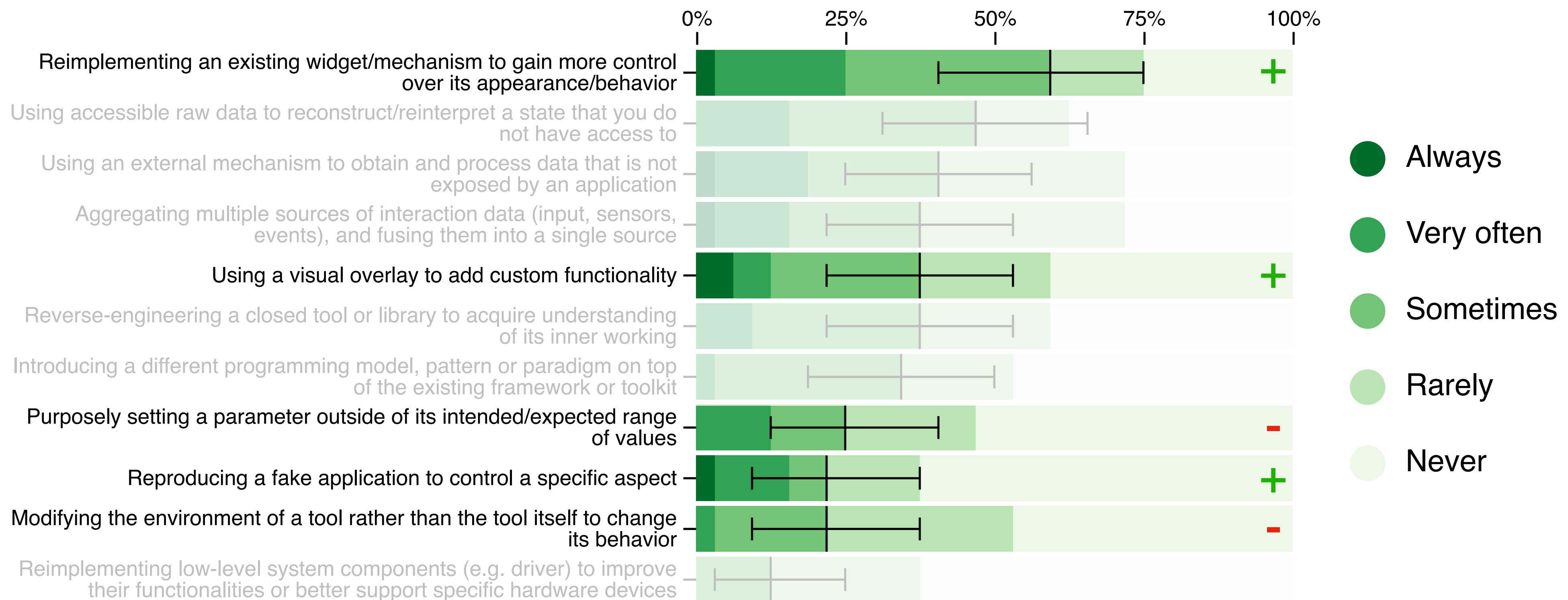




# Interviews & Survey

## Observation 5

*Researchers will often implement new features from scratch rather than patch existing applications or widgets*



How can we design or adapt existing frameworks  
and toolkits to support researchers?

# Design recommendations

## Related work

### Rationales from toolkits:

- rarely discussed in papers
- highly contextual
- lack of justifications on positive impacts

### Rationales from frameworks:

- highly abstract
- no general consensus
- lack of tradeoffs acknowledgement

### Programming requirements studies:

- good to understand the complexity of frameworks
- need more traction to generate more in-depth descriptions

# Design recommendations

## Influencing frameworks

How can we have a good impact on frameworks/toolkits?

- code artefact (plugin, toolkit)
- *usage study*
- tech talk (e.g. Qt World Summit, Android Dev Summit)
- join/create a working group
- *design principles*

Duplicate, Accumulate, Defer (DAD)



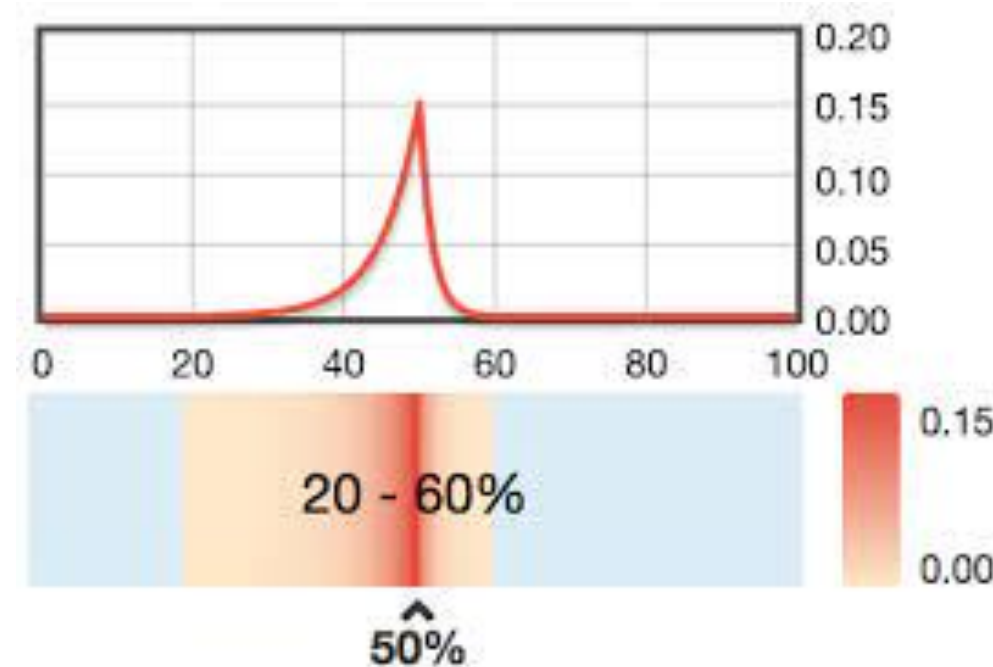
# Design recommendations

## Duplicate

*Allow the duplication of singular elements to foster opportunities for extensions*

Method: for each element/property/argument

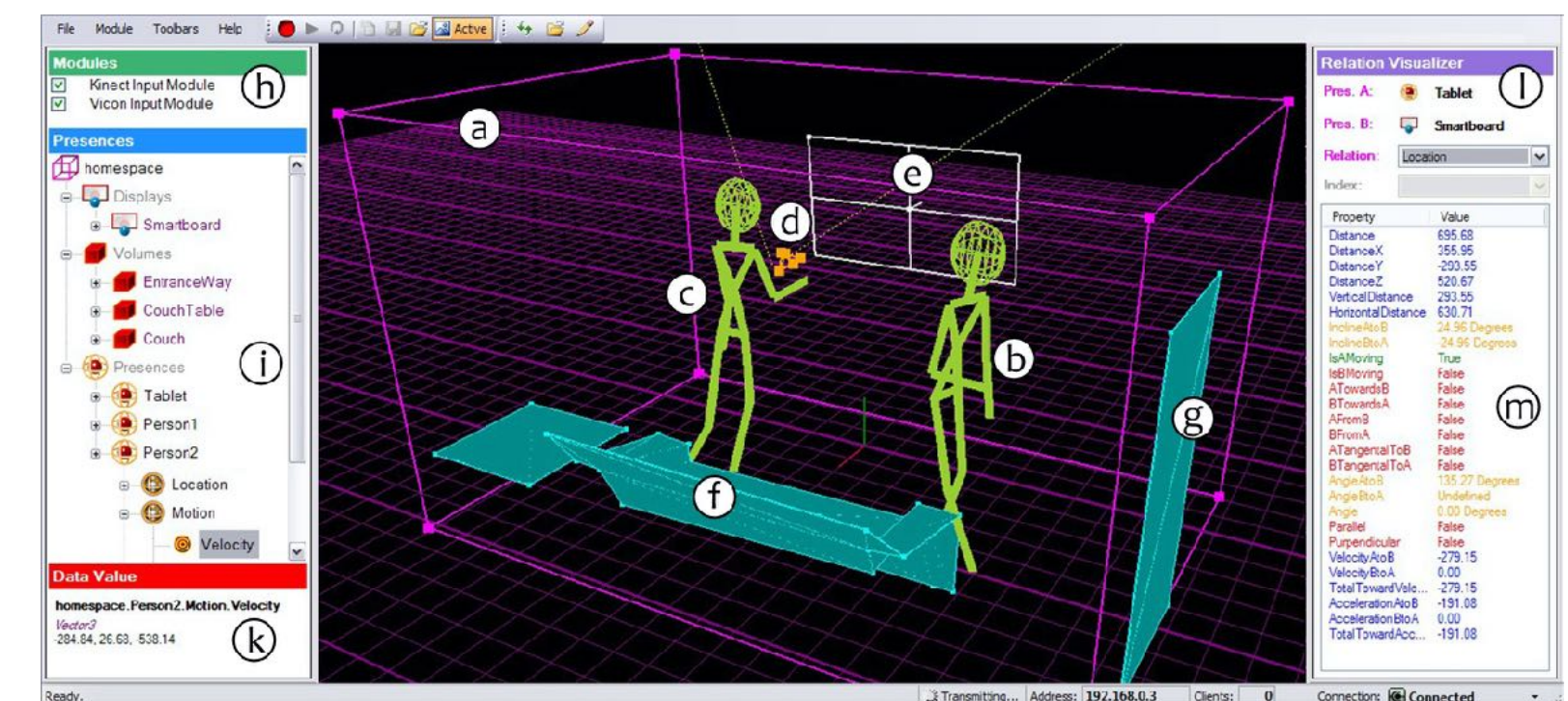
- 1) Is it expected to be unique?
- 2) Could it make sense to allow many?



Probability Distribution Sliders (Greis et al.)



ExposeHK (Malacria et al.)



Proximity Toolkit (Marquardt et al.)

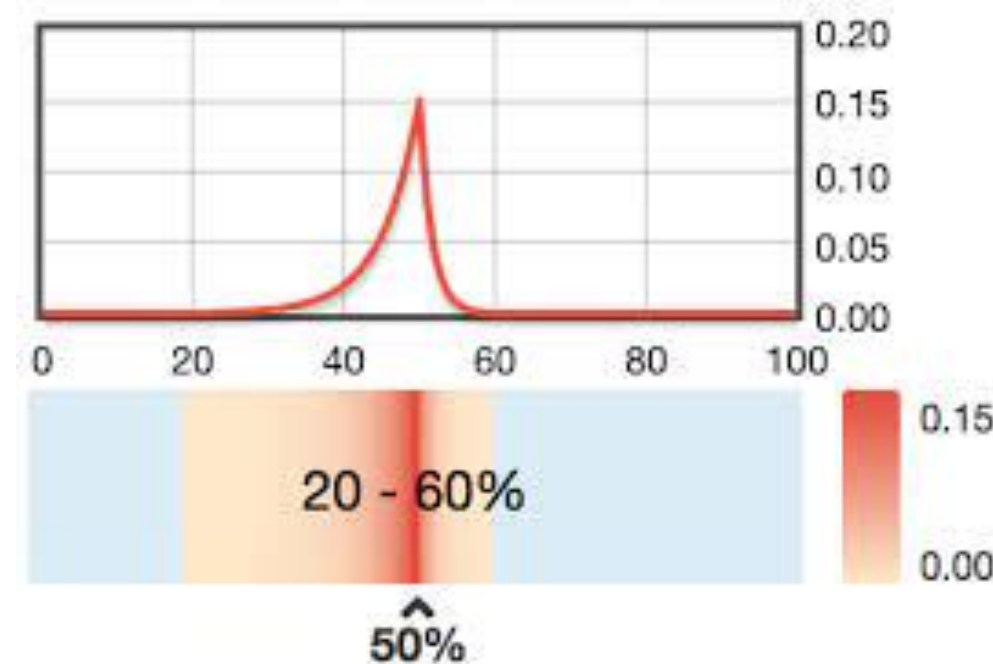


# Design recommendations

Duplicate

Do not implement these examples → finer reuse/composition

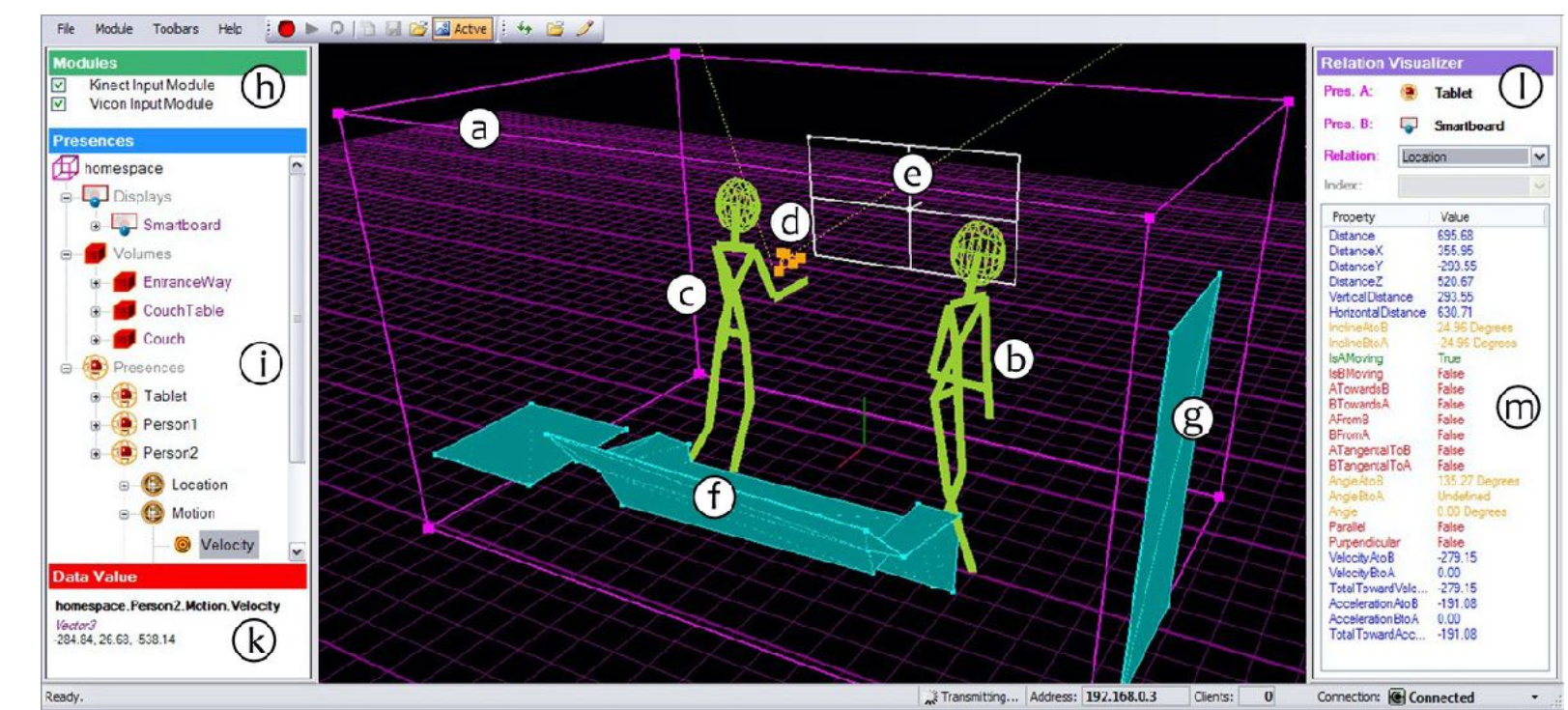
Hard support → toolkits (e.g. multiple mice → libpointing)



Probability Distribution Sliders (Greis et al.)



ExposeHK (Malacria et al.)



Proximity Toolkit (Marquardt et al.)



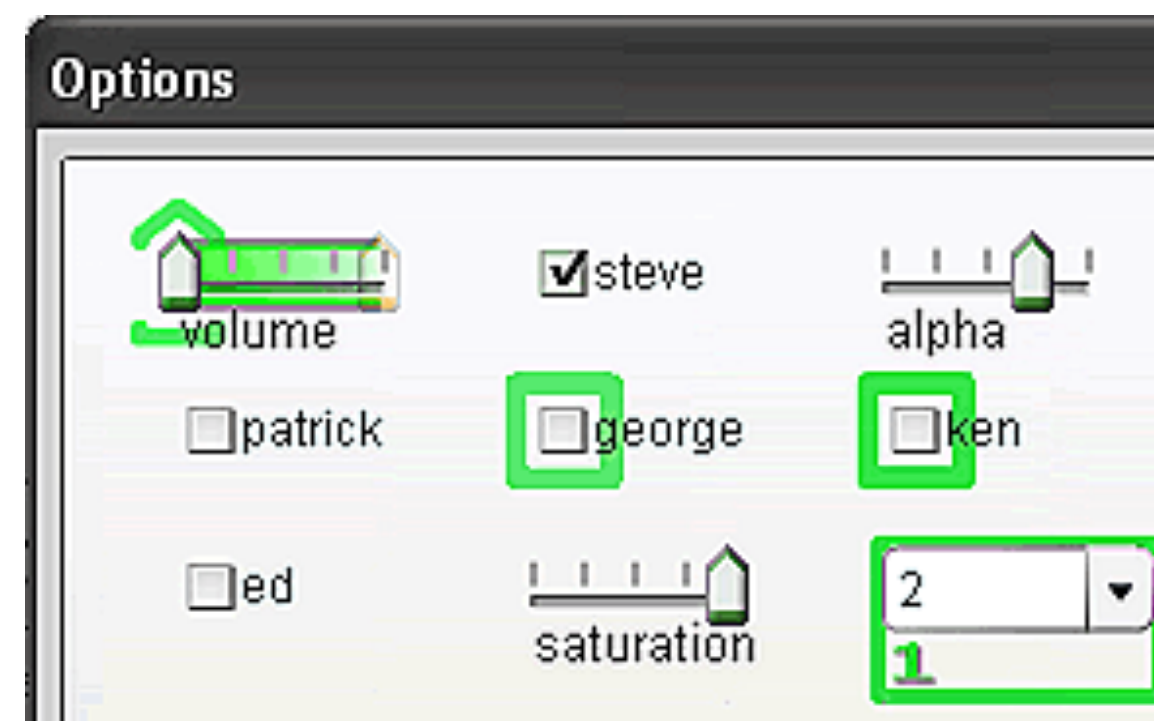
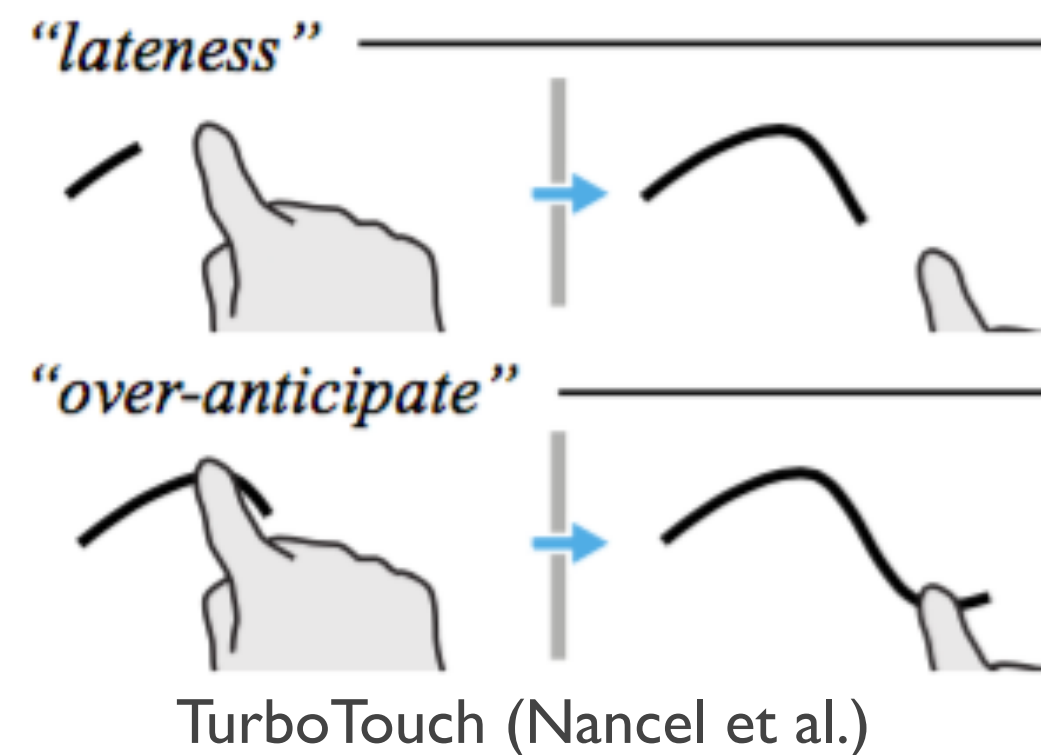
# Design recommendations

## Accumulate

*Accumulate rather than replace to keep a history of changes*

Method: for each property/argument

- 1) Is this data replaced by another?
- 2) Could it make sense to keep both at any time?



Phosphor (Baudisch et al.)



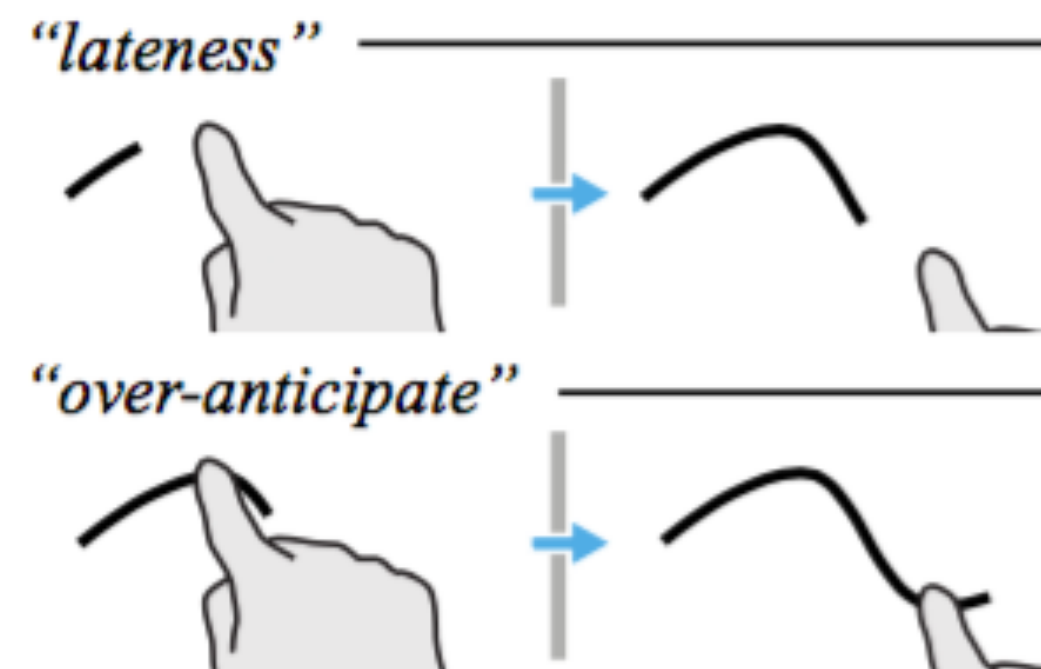
ForceEdge (Antoine et al.)

# Design recommendations

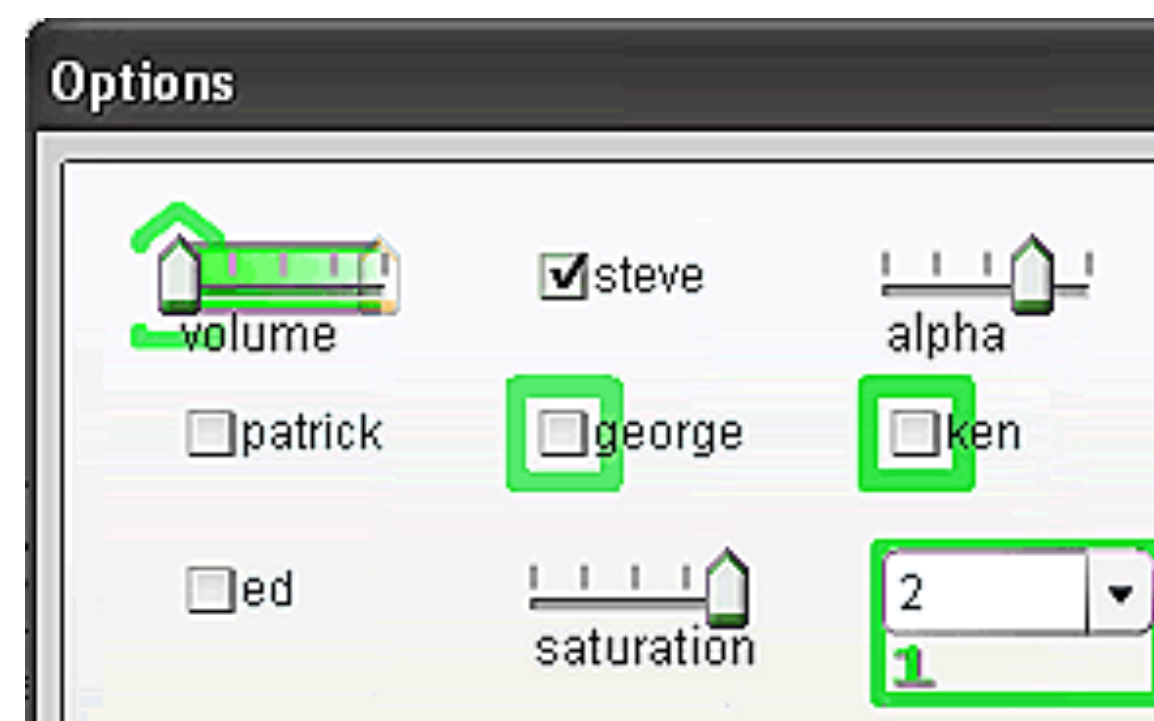
Accumulate

Accumulation over time/space

Polymorphism



TurboTouch (Nancel et al.)



Phosphor (Baudisch et al.)



ForceEdge (Antoine et al.)

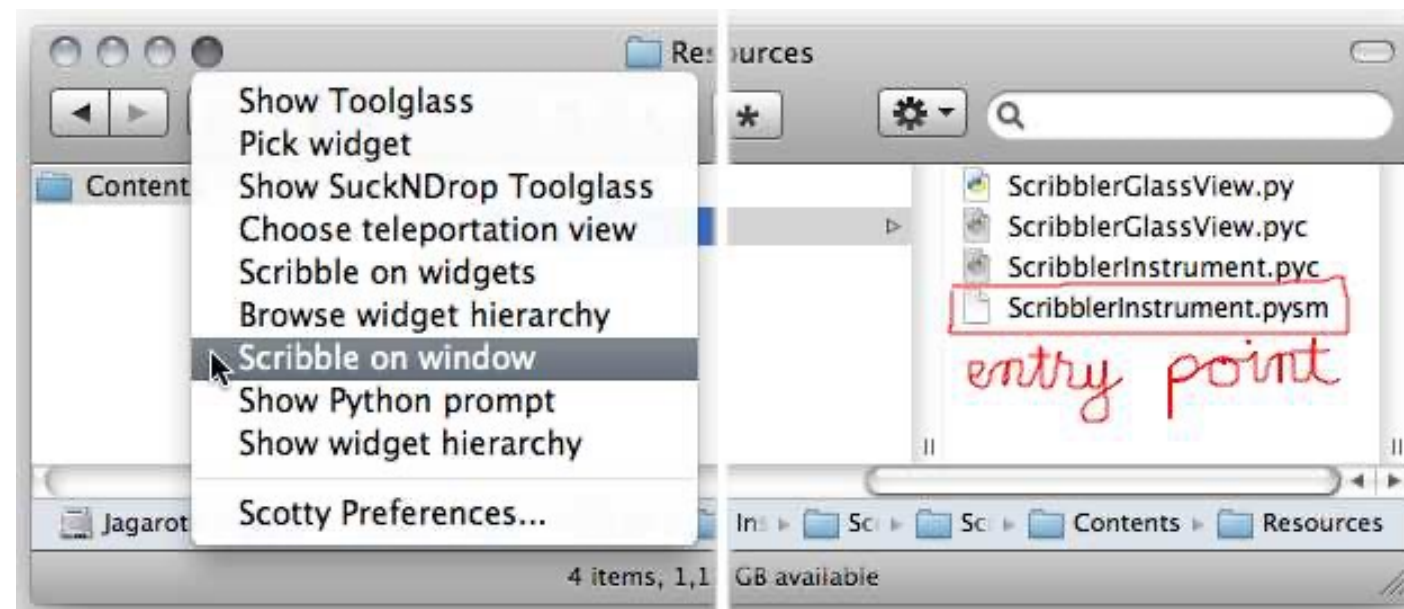
# Design recommendations

# Defer

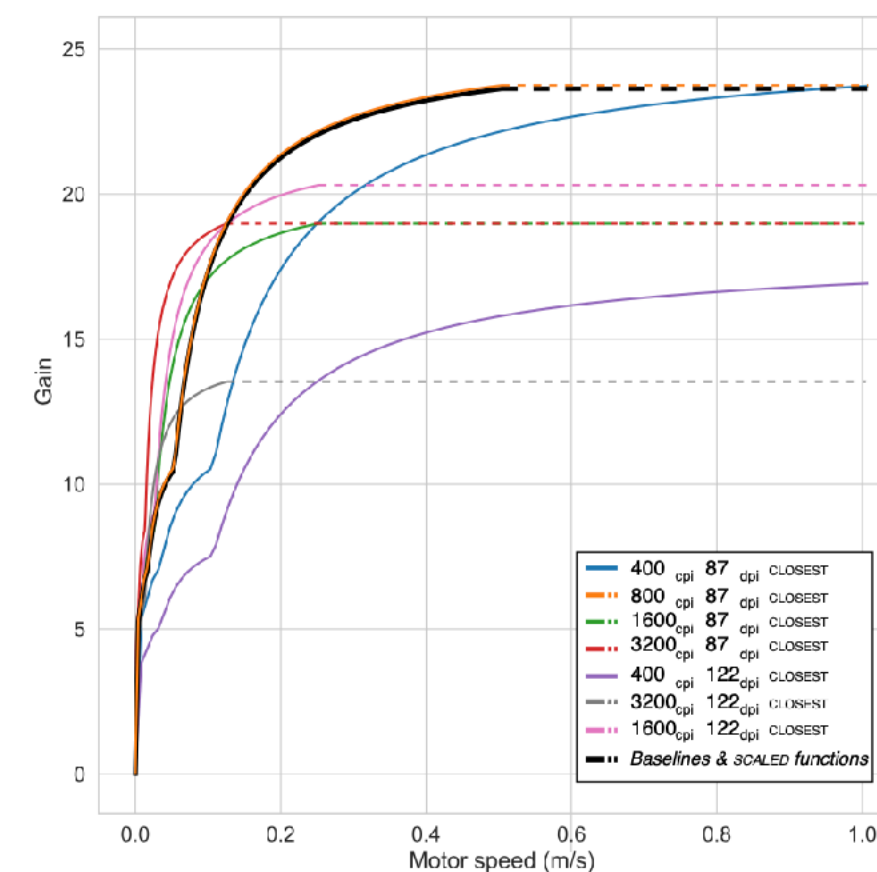
*Defer the execution of predefined behaviors to enable their monitoring and replacement*

## Method: for each function/method

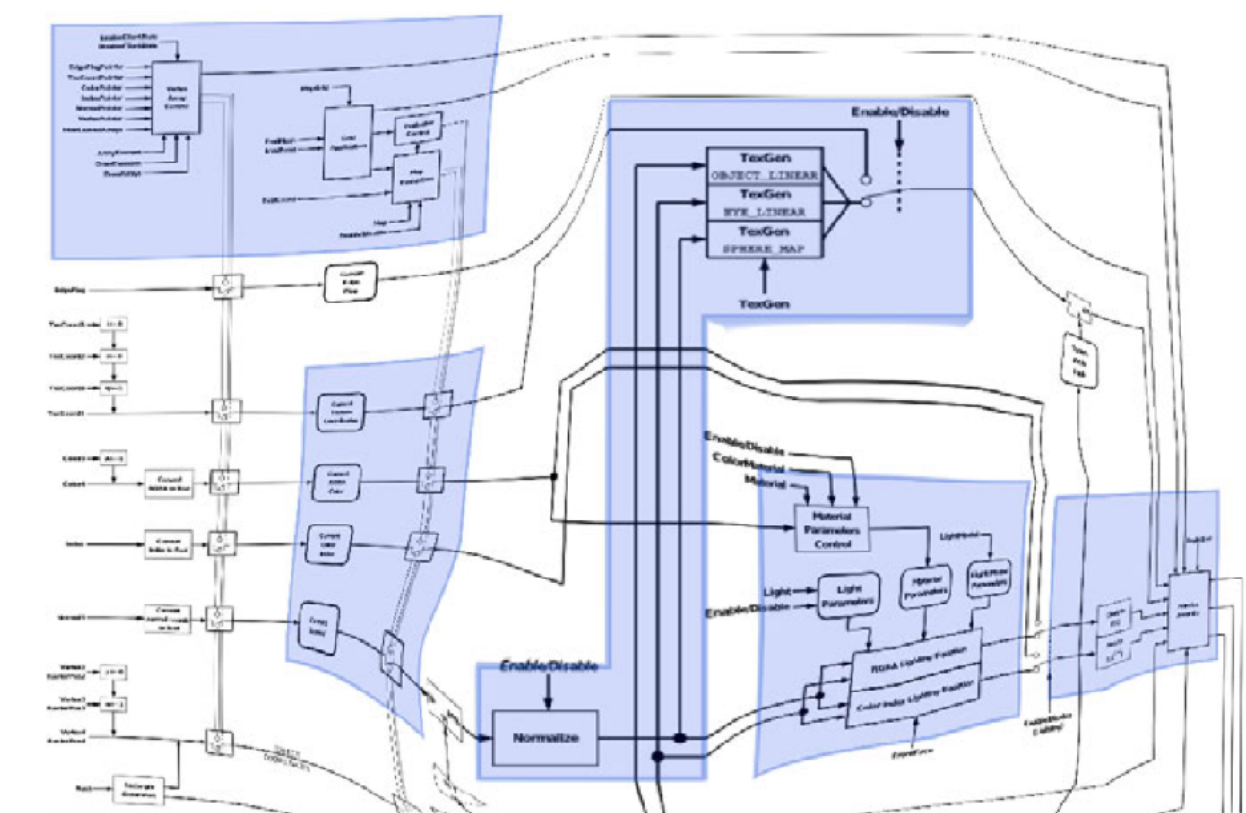
- 1) Can this action be intercepted? (i.e. canceled, altered or repeated)
- 2) If not, could it be useful at run-time or compile-time?



## Scotty (Eagan et al.)



libpointing (Casiez et al.)



JellyLens (Pindat et al.)



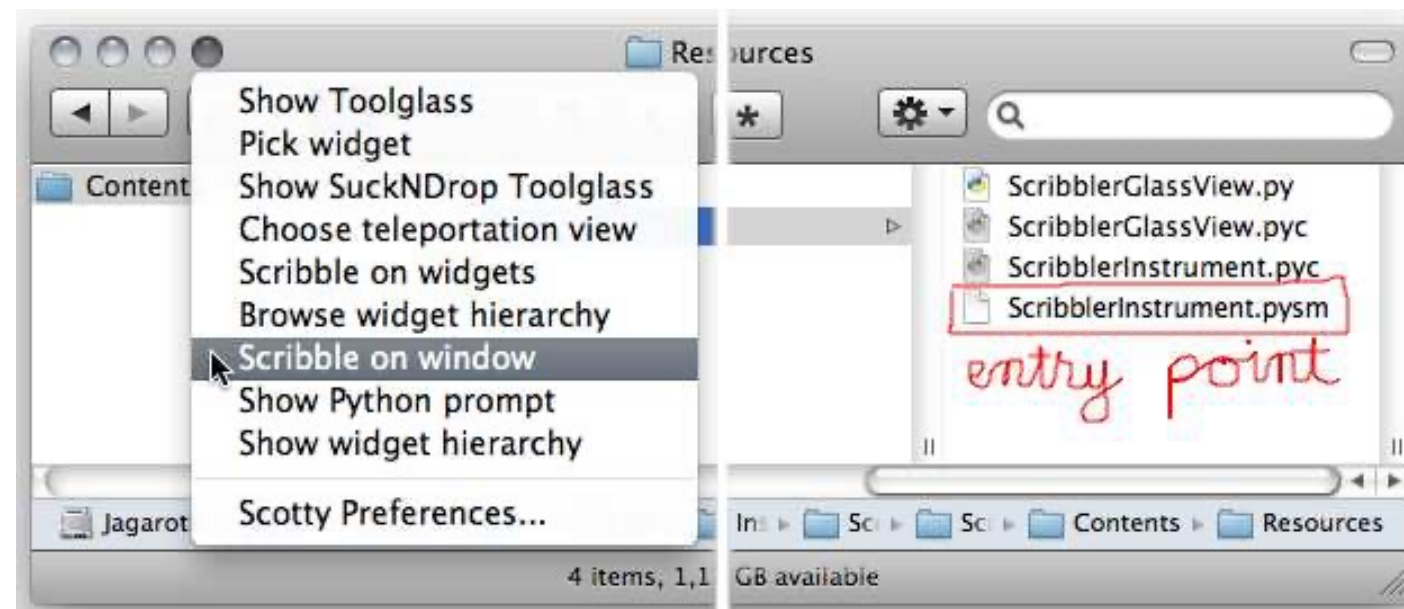
# Design recommendations

Defer

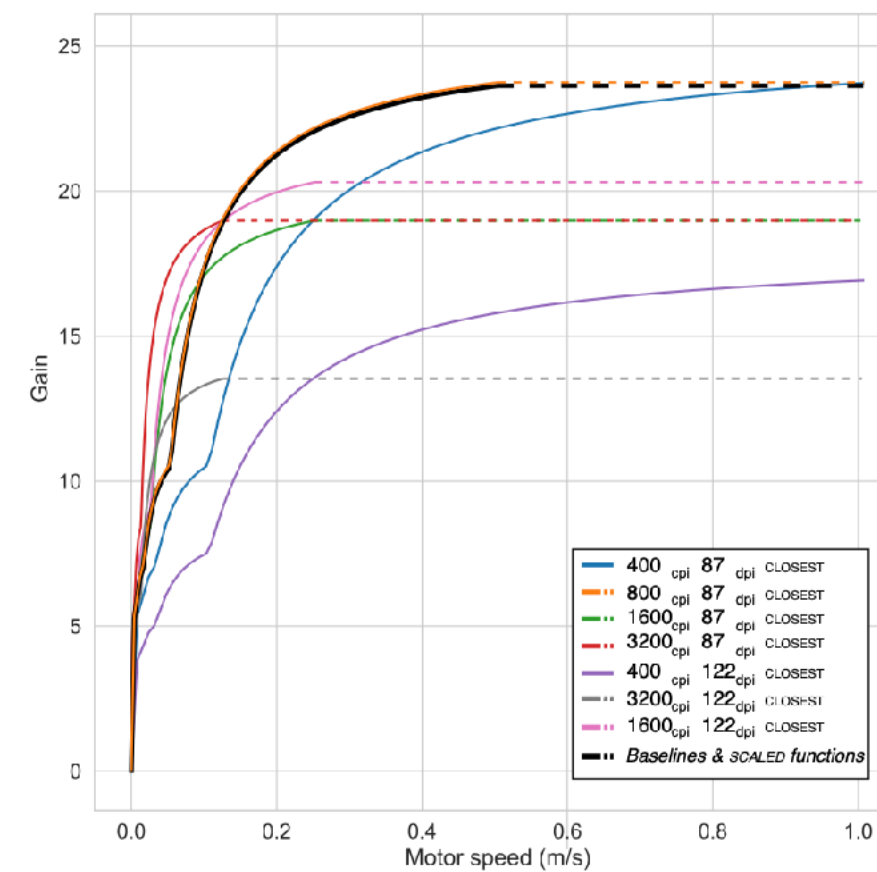
Split commands into (i) placing an order and (ii) executing it

More scalable indirection mechanisms:

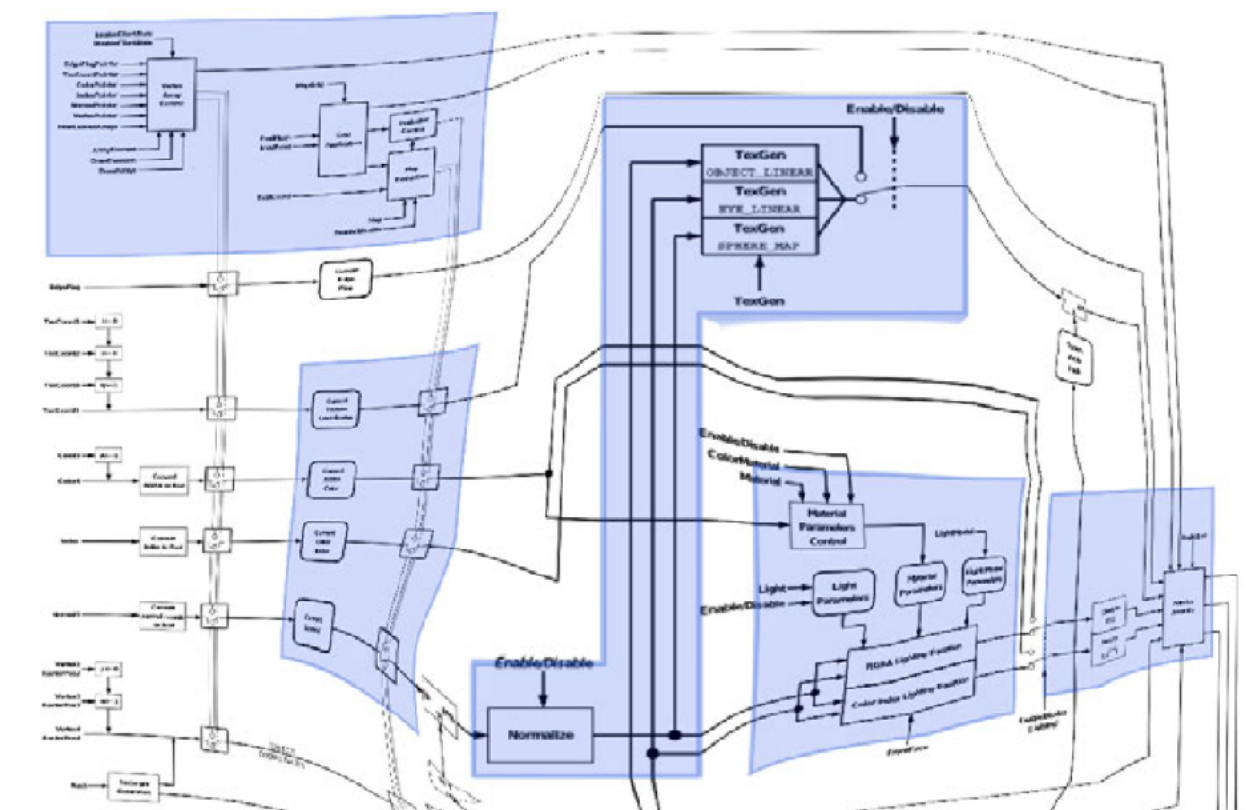
- open intermediate structures (e.g. DOM, framebuffer)
- software buses



Scotty (Eagan et al.)



libpointing (Casiez et al.)



JellyLens (Pindat et al.)

# Conclusion and future work

## Contributions:

- key observations about researchers when programming novel interaction techniques
- design principles to better support them in frameworks & toolkits

## Future work:

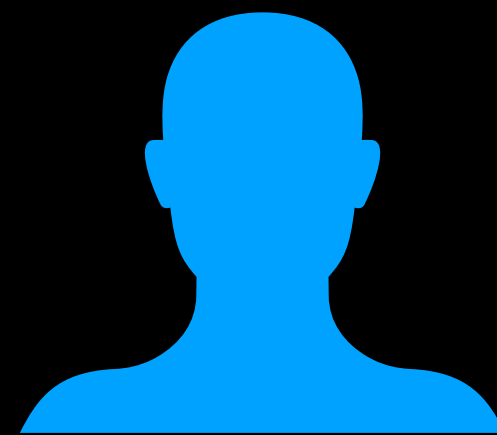
- promoting these principles
- classify programming practices vs types of interaction techniques

Thank you for your attention

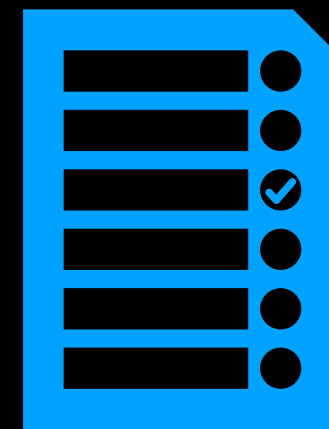


# Interviews & Survey

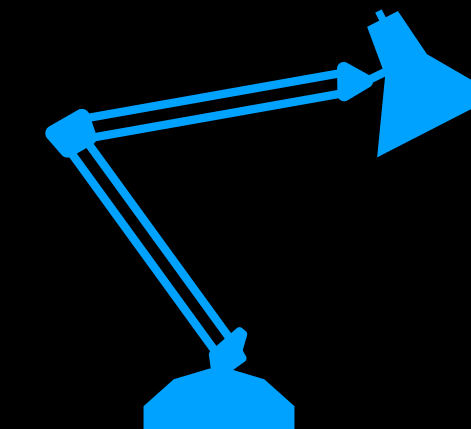
## Interviews



9 HCI researchers (+1 pilot)  
6 Seniors researchers  
1 Engineer  
1 PhD student  
1 Master student



Semi-structured  
2~4 past projects  
Problems faced  
Typical development cycle

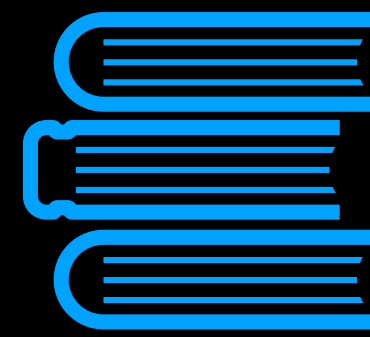


in-situ interviews  
1 interviewer, 1 participant  
audio recording



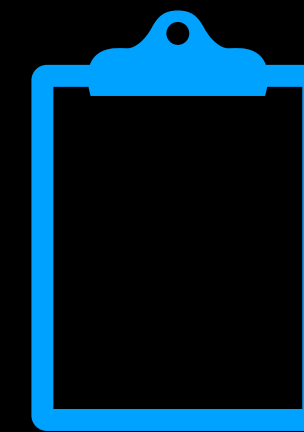
audio  
9h39

transcription



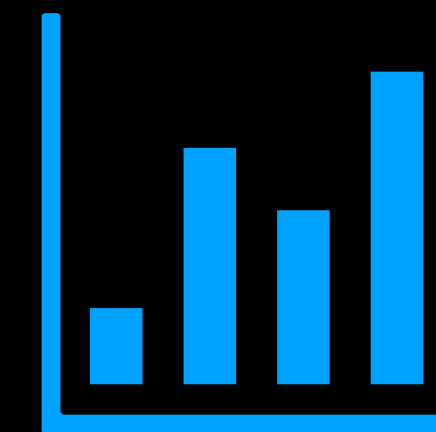
2519 lines  
9 files

extraction



228 codes  
• problems  
• strategies  
• utilities  
• needs

analysis



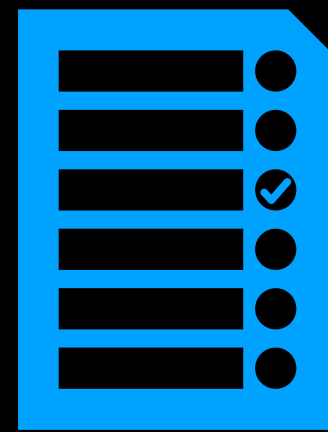
48 themes  
• problems  
• strategies  
• utilities

# Interviews & Survey

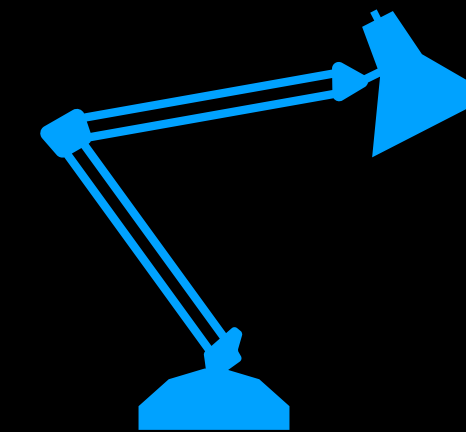
## Survey



32 participants (+4 pilot)  
2/3 code < 40% of their time  
2/3 advanced or expert



online questionnaire  
rating relevance of items  
20 minutes



[chi-announcements@acm.org](mailto:chi-announcements@acm.org)  
2nd batch to former teams

- What are the most important criteria for choosing interaction libraries? (R1)
- What are the most limiting implementation problems for researchers? (R2)
- Which strategies are most used to circumvent and overcome these problems? (R3)

# Interviews & Survey

## Limits & Scope

Interviews with local team of researchers → risk of missing some problems/strategies

Being familiar with the interviewees → risk of overestimating the severity of problems

Lack of baseline survey with non-researchers → lack of emphasis on the uniqueness of research needs

Scope: understand why researchers are unsatisfied & suggest directions of improvement

# Design recommendations

## Rationales from toolkits

Example in D3 (Bostock et al., 2011):

- when a scene is generated from data, specify explicit transformations rather than letting the scene be generated implicitly
- the update of a property depending on another is immediate rather than deferred to facilitate live inspection and debugging
- intermediate representations rely on existing native formats to leverage existing user knowledge and helper tools

Extracting recommendations for other frameworks/toolkits:

- rarely discussed in papers
- highly contextual
- lack of justification on their positive impact for users

TOOLKIT	VENUE	YEAR	REF	TYPE	1	2	3	4
1 Canvas Toolkit	CHI	1999	148					
2 DIERK	CHI	2005	143					
3 Toolkit Level Support for Ambiguity in Recognition-Based Interfaces	CHI	2002	343					
4 SALTIN	CHI	2004	148					
5 Phidgits	CHI	2004	143					
6 GYD	CHI	2003	40					
7 Synkassy	CHI	2003	36					
8 DRIFF	CHI	2003	36					
9 MAUI Groupware Toolkit	CHI	2004	451					
10 DiamondGrip	CHI	2004	391					
11 Paper Midge	CHI	2004	343					
12 Calder	CHI	2004	343					
13 ECDN	CHI	2004	343					
14 Toolkit Design for Interactive Structured Graphics	CHI	2004	343					
15 DARE	CHI	2004	343					
16 Adaptive Post WIMP Toolkit	CHI	2004	343					
17 Peripheral Displays Toolkit	CHI	2004	343					
18 Pichart	CHI	2004	343					
19 Subchart	CHI	2004	343					
20 d-Tools	CHI	2004	343					
21 SwingStates	CHI	2004	343					
22 Exemplar	CHI	2004	343					
23 CapToolKit	CHI	2004	343					
24 ReactVision	CHI	2004	343					
25 Shared Phidgits	CHI	2004	343					
26 VoodonO	CHI	2004	343					
27 Kromay	CHI	2004	343					
28 Damaak	CHI	2004	343					
29 VoodonSketch	CHI	2004	343					
30 GYD	CHI	2004	343					
31 PACT	CHI	2004	343					
32 Proavis	CHI	2004	343					
33 Dikali	CHI	2004	343					
34 Imulino	CHI	2004	343					
35 Amantio Toolkit	CHI	2004	343					
36 Intelligence Toolkit	CHI	2004	343					
37 D-MACS	CHI	2004	343					
38 Pichart	CHI	2004	343					
39 JouchID Toolkit	CHI	2004	343					
40 D3	CHI	2004	343					
41 Fluidity Toolkit	CHI	2004	343					
42 Phylota	CHI	2004	343					
43 d-MultiTouch	CHI	2004	343					
44 Pichart	CHI	2004	343					
45 NET Designer	CHI	2004	343					
46 HapticTouch Toolkit	CHI	2004	343					
47 Midos	CHI	2004	343					
48 PaperBus	CHI	2004	343					
49 Wicket	CHI	2004	343					
50 KinectArm	CHI	2004	343					
51 OpenCageSense	CHI	2004	343					
52 ToyValon Toolkit	CHI	2004	343					
53 Saver	CHI	2004	343					
54 PictKama	CHI	2004	343					
55 XE Studio	CHI	2004	343					
56 XE Studio	CHI	2004	343					
57 PolyChrome	CHI	2004	343					
58 PaperPulse	CHI	2004	343					
59 WatchConnect	CHI	2004	343					
60 Waves	CHI	2004	343					
61 Solid Toolkit	CHI	2004	343					
62 C4	CHI	2004	343					
63 Makers Marks	CHI	2004	343					
64 Playdirt	CHI	2004	343					
65 Racerlap	CHI	2004	343					
66 Let Your Body Move Toolkit	CHI	2004	343					
67 CircuitStark	CHI	2004	343					
68 EngleStar	CHI	2004	343					
69 Finalat	CHI	2004	343					

Evaluation Strategies for HCI Toolkit Research  
(Ledo et al.)

# Design recommendations

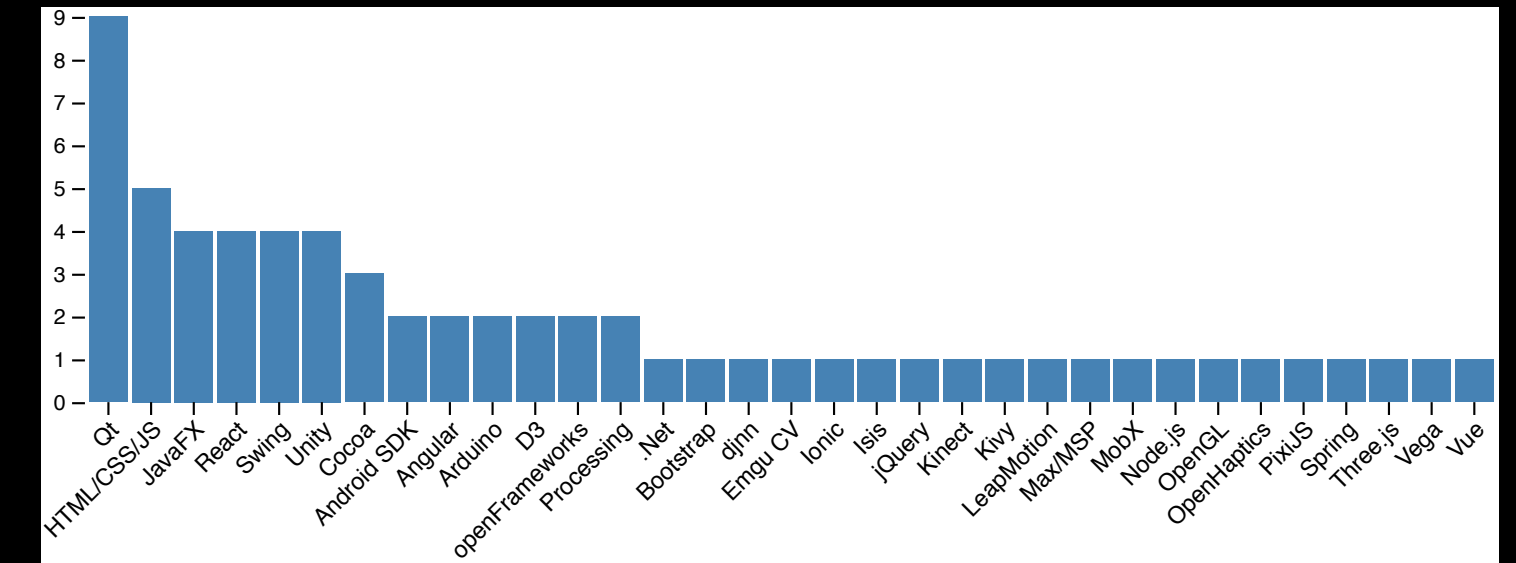
## Rationales from frameworks

Example from Qt (Knoll, 2017):

- APIs that lead to readable and maintainable code
- easy to learn and use but hard to misuse
- performant
- flexible
- keeping it simple
- API stability
- world class tools

Extracting recommendations for other frameworks/toolkits:

- highly abstract
- no general consensus
- lack of tradeoffs acknowledgement



# Design recommendations

## Studies on researchers' needs

Example in *Usability requirements for interaction-oriented development tools* (Letondal, 2010):

- minimising information complexity
- minimising access complexity
- minimising unpredictability
- graphics
- runtime adaptation
- interaction modalities
- distribution
- supporting code production
- matching code and execution
- managing the life cycle
- managing reuse and knowledge capitalization
- managing collective development

Extracting recommendations for other frameworks/toolkits:

- good for understanding complexity of frameworks and comparing them
- need more traction to generate more in-depth descriptions