
# Understanding Asynchronous Interactions In Full-Stack JavaScript

Saba Alimadadi, Ali Mesbah and Karthik Pattabiraman

ICSE 2016

saba@ece.ubc.ca

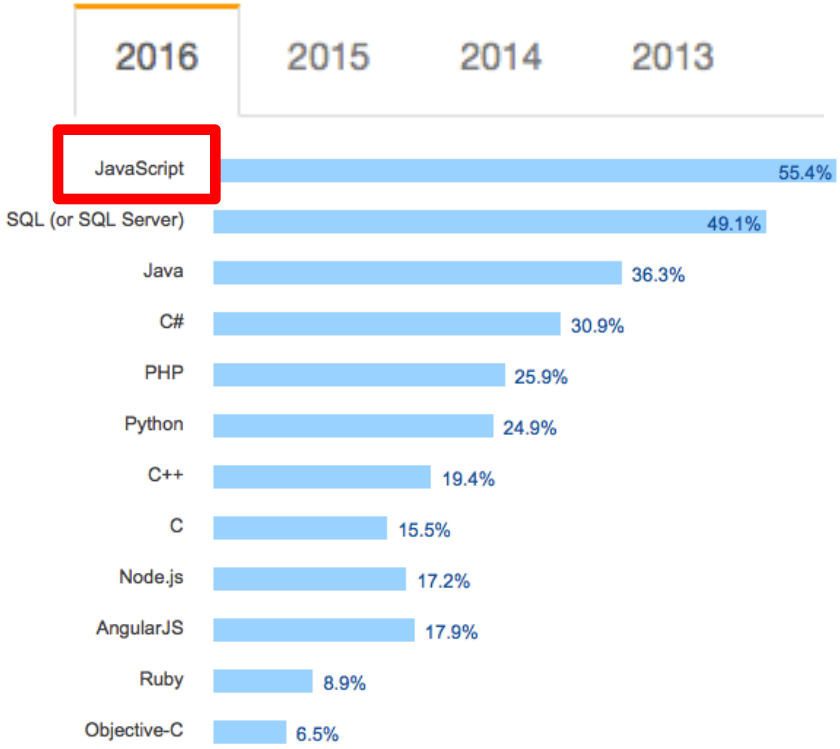
Sahand: http://github.com/saltlab/sahand



a place of mind
THE UNIVERSITY OF BRITISH COLUMBIA

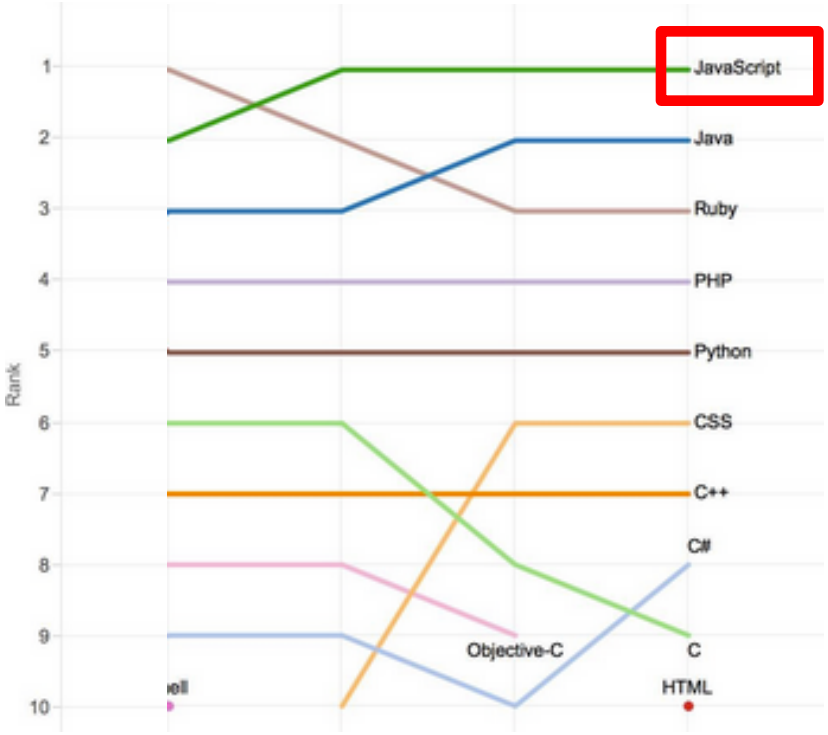
stackoverflow

## **JavaScript**:
## Most popular language

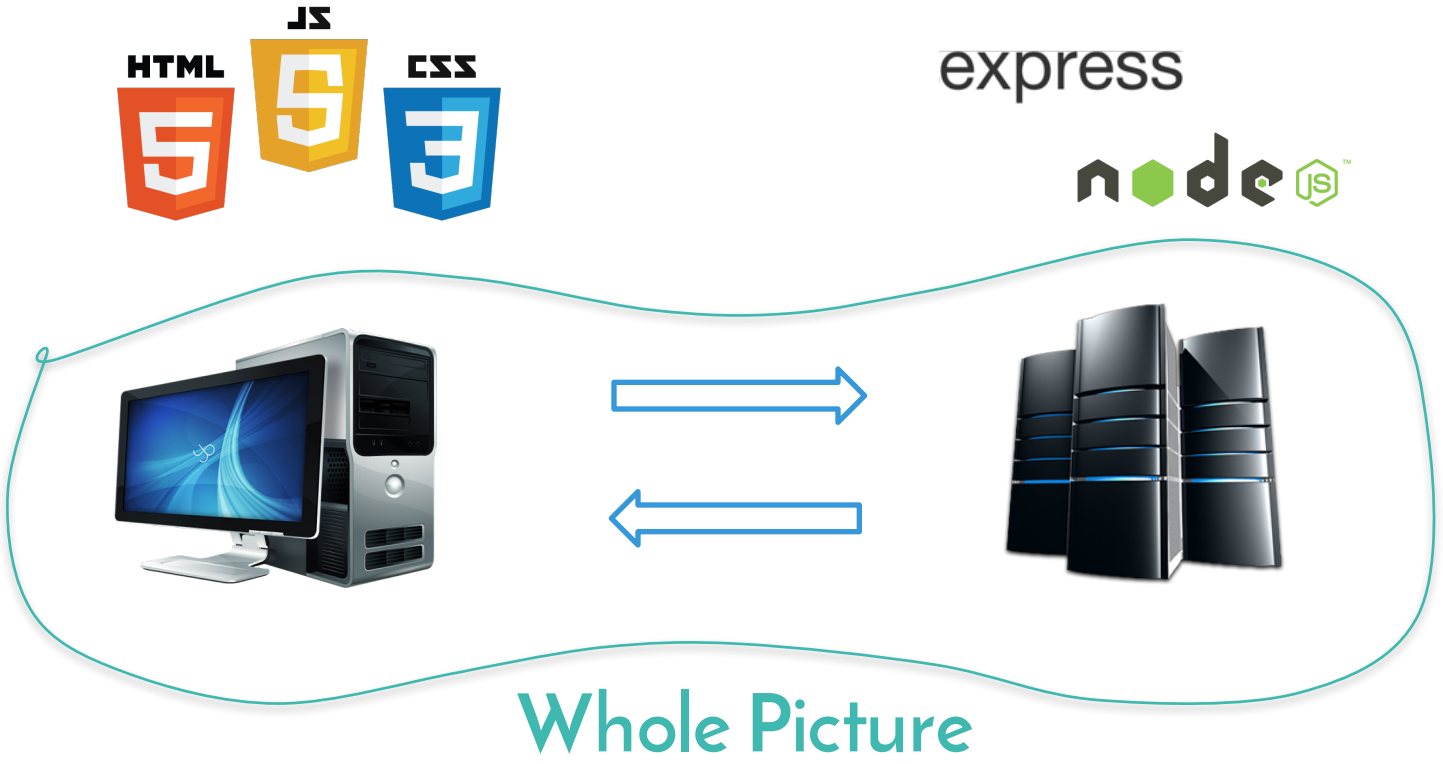



GitHub

## **JavaScript**:
## Top languages on GitHub



# Understanding JavaScript Apps

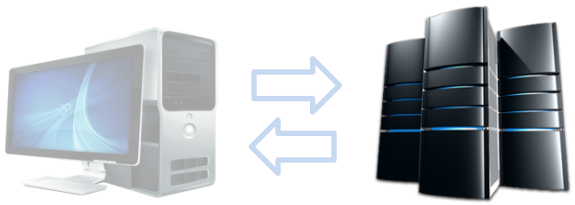


# Challenge 1. **Server**-Side Callbacks

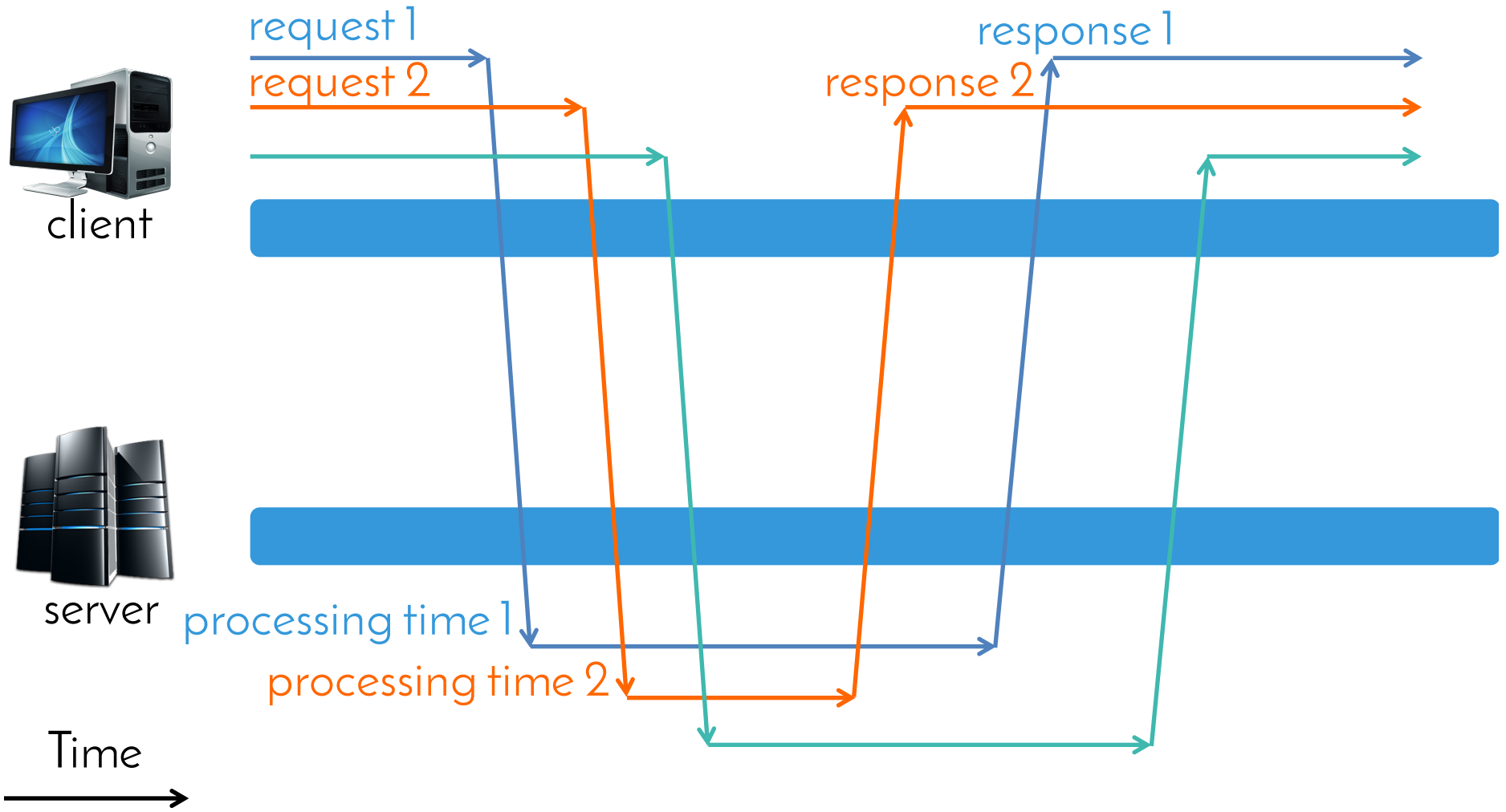
- Asynchronous execution
- Callback hell

Little pyramid of **doom**

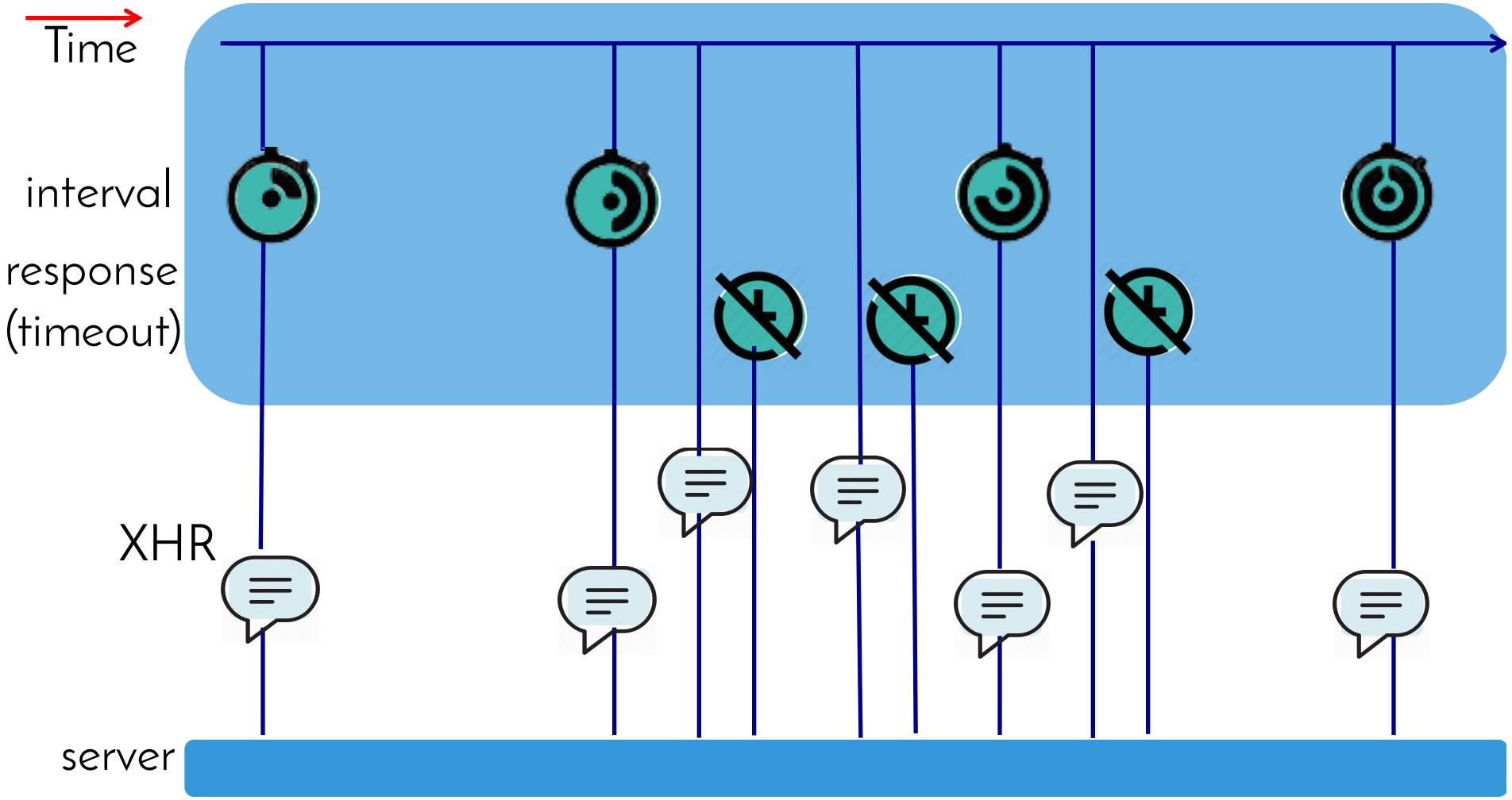
```
fs.readdir(source, function(err, files) {
  files.forEach(function(filename, fileIndex)
    gm(source + filename).size(function(err, values) {
      widths.forEach(function(width, widthIndex) {
        this.resize(w, h).write(newName, function(err) {
        })
      })
    })
  })
}) // example from callbackhell.com
```



# Challenge 2. **Network** Communications



# Challenge 3. Asynchronous **Client** Side



# Summary of Challenges

- Server-side callbacks
- Network communication
- Asynchronous client side

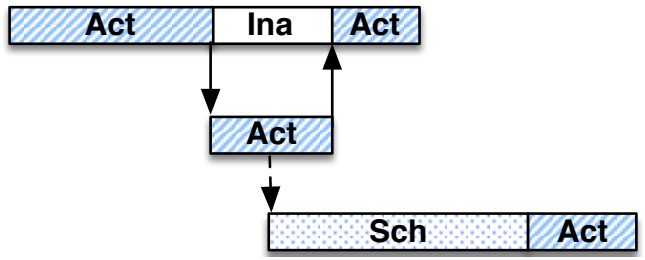
Related work:

| | | |
|---|---|---|
| Zaidman et al. EMSE'13 | Hibschman et al. UIST'14 | Alimadadi et al. ICSE'14, ECOOP'15 |

# Our Approach: *Sahand*

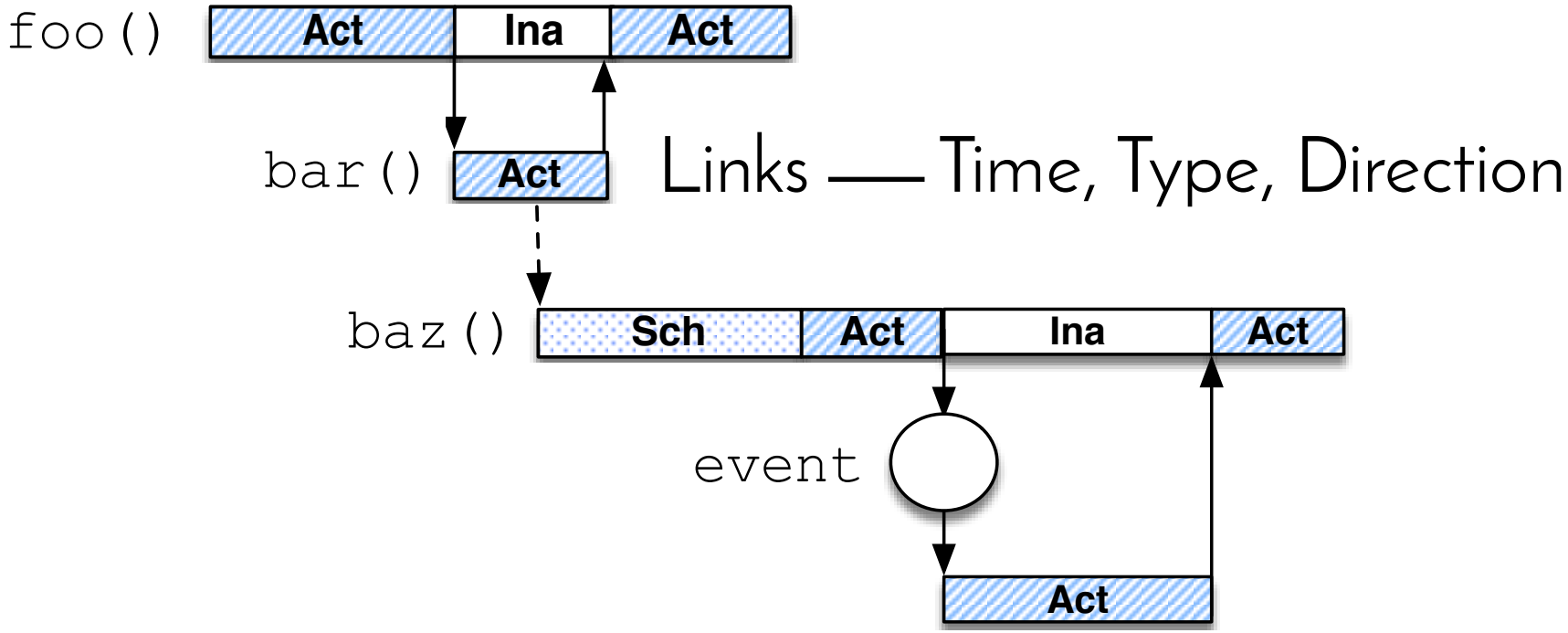
Application

1. Instrument automatically
2. Trace full-stack execution
3. Infer a behavioural model
4. Visualize the model

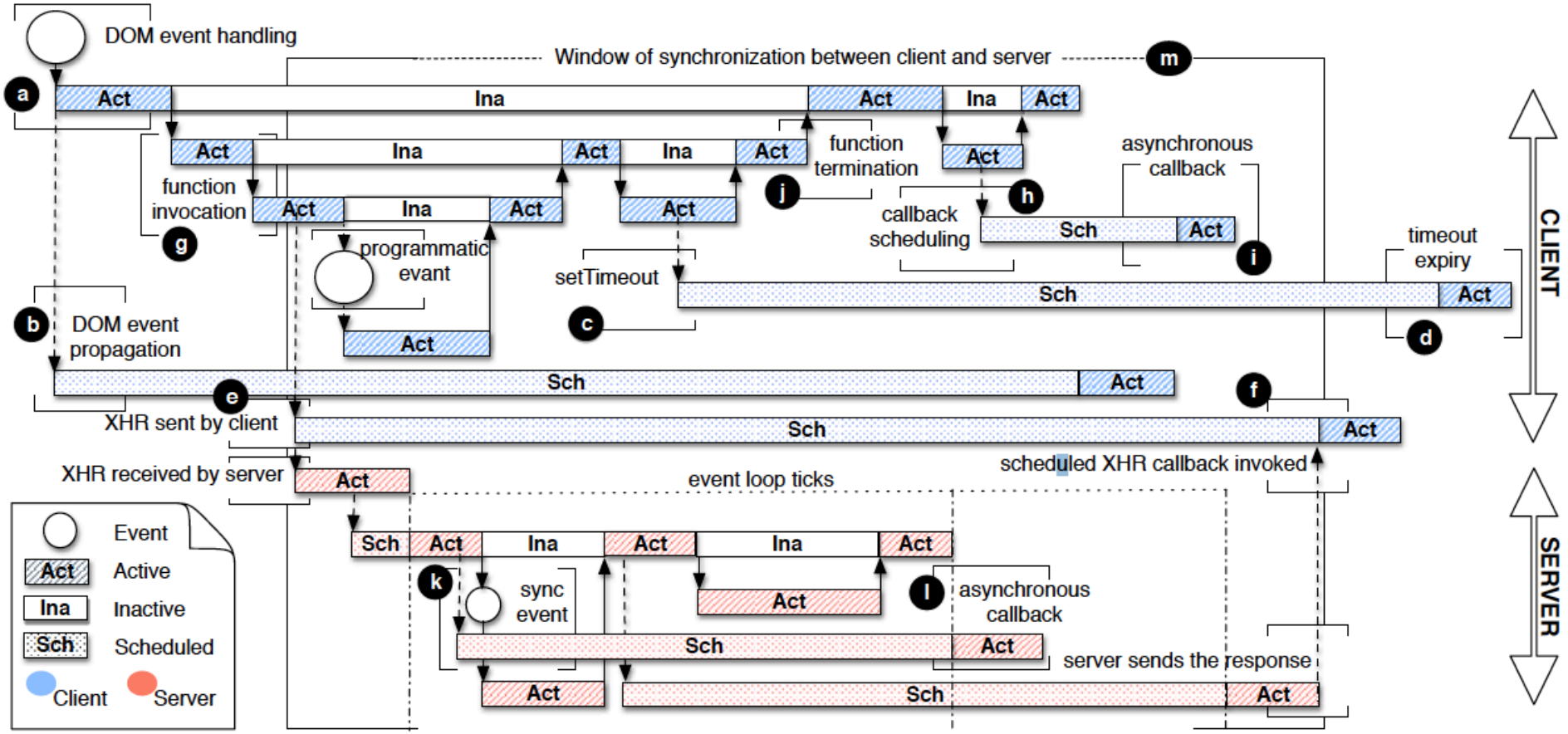


# Behavioral Model

Nodes
- Lifelines of function executions
- (A)Synchronous client/server events



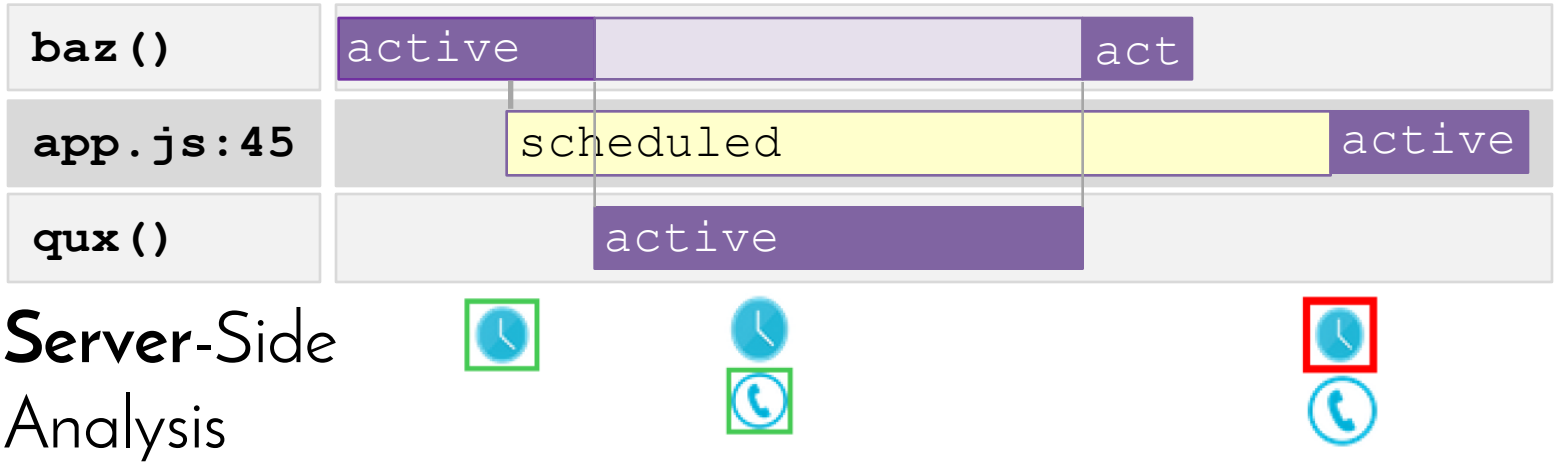
# Real Behavioural Models Are Complex



# Visualization

**Client**-Side Analysis

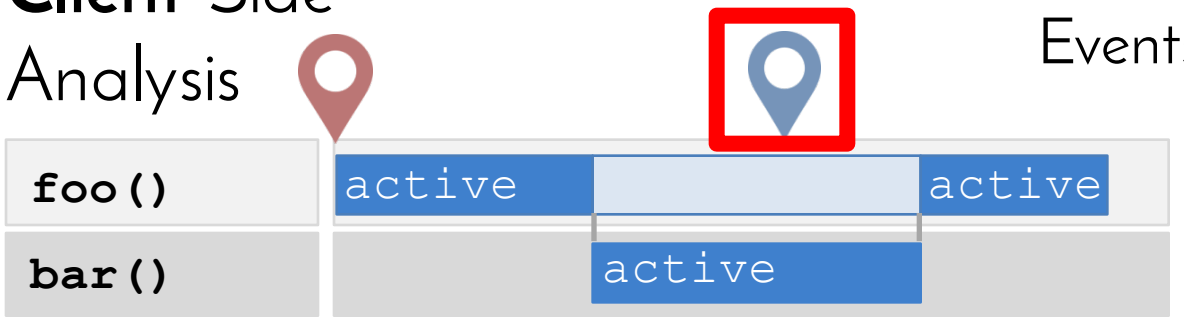

Connecting client and server




**Server**-Side Analysis

# Visualization

**Client**-Side Analysis

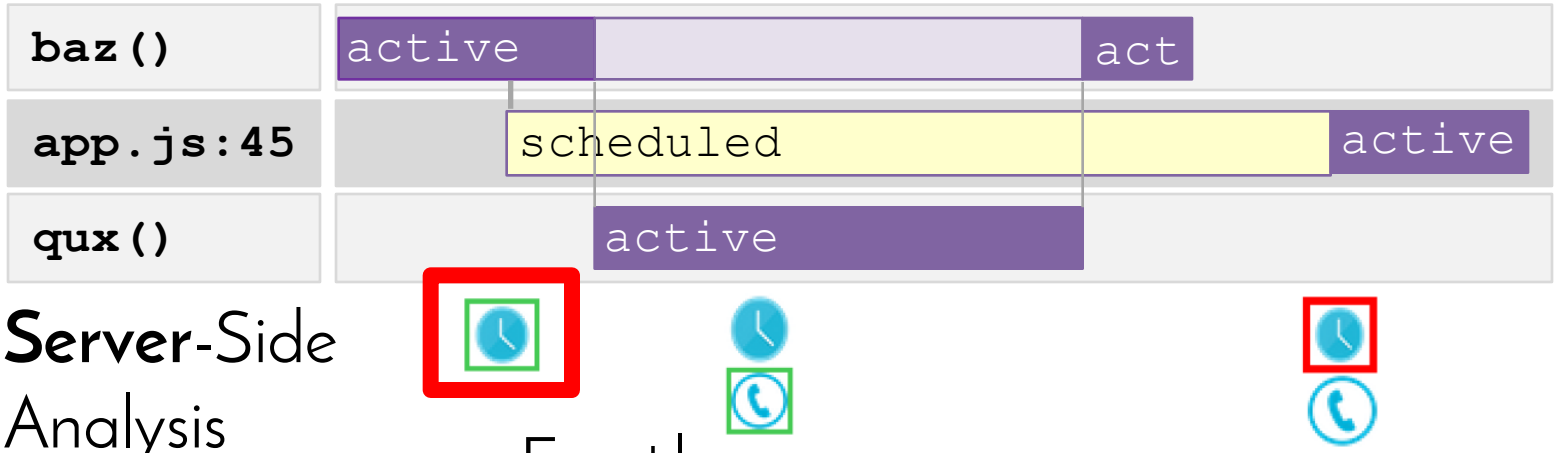



Events and DOM interactions

Timeouts

XHRs

Time —Temporal primitives→ Time points

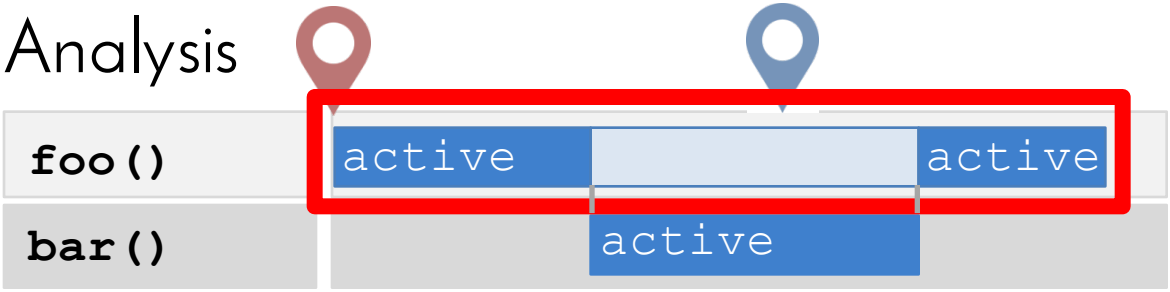



**Server**-Side Analysis

Event loop

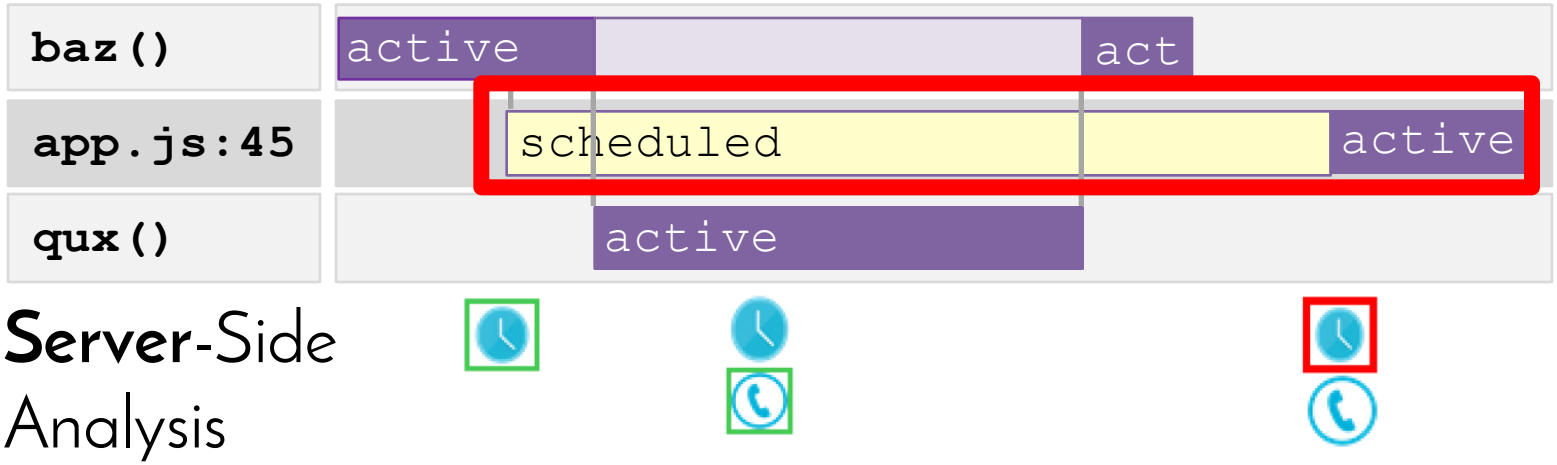
# Visualization

**Client**-Side Analysis




Function executions

Time —Temporal primitives→ Time intervals

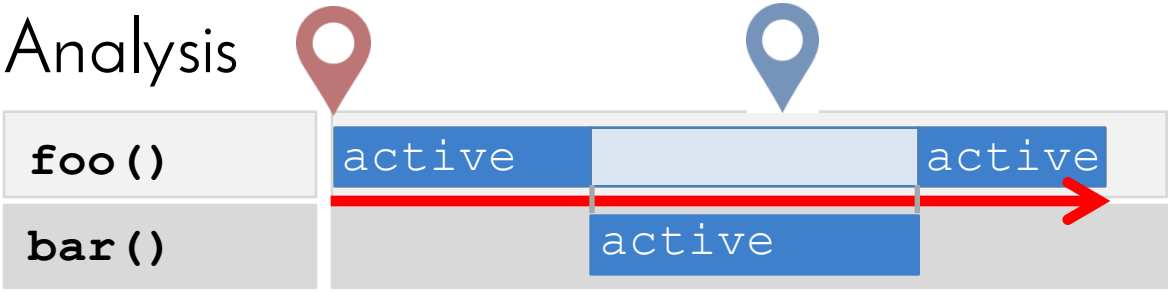



Callbacks

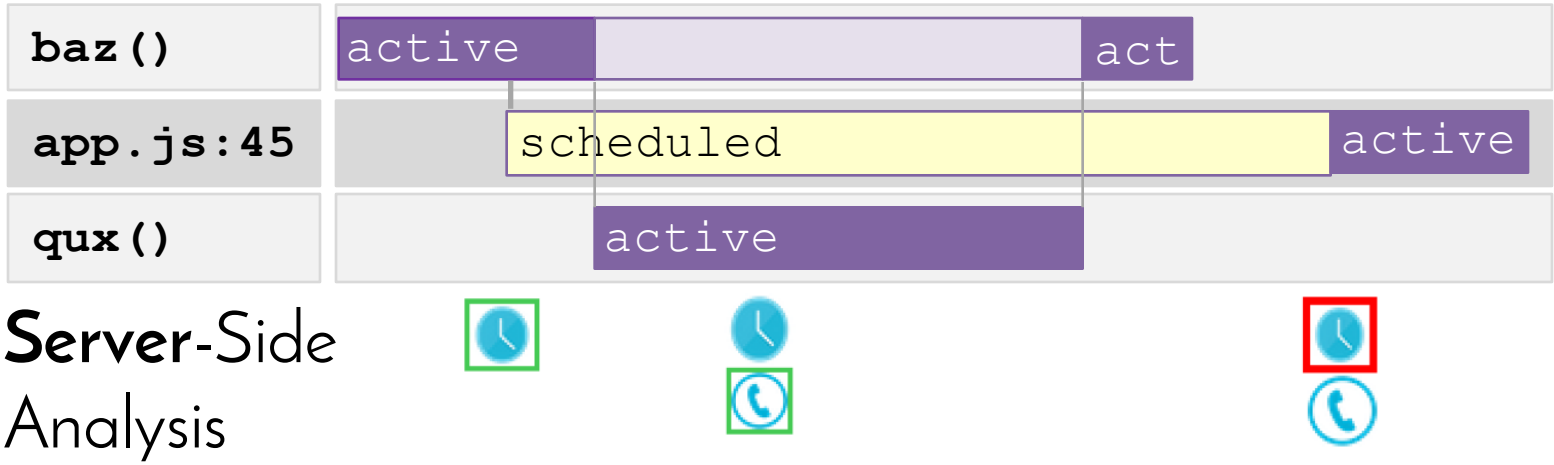
**Server**-Side Analysis

# Visualization

**Client**-Side Analysis




Time ⟶ Structure of time ⟶ Linear & Branching

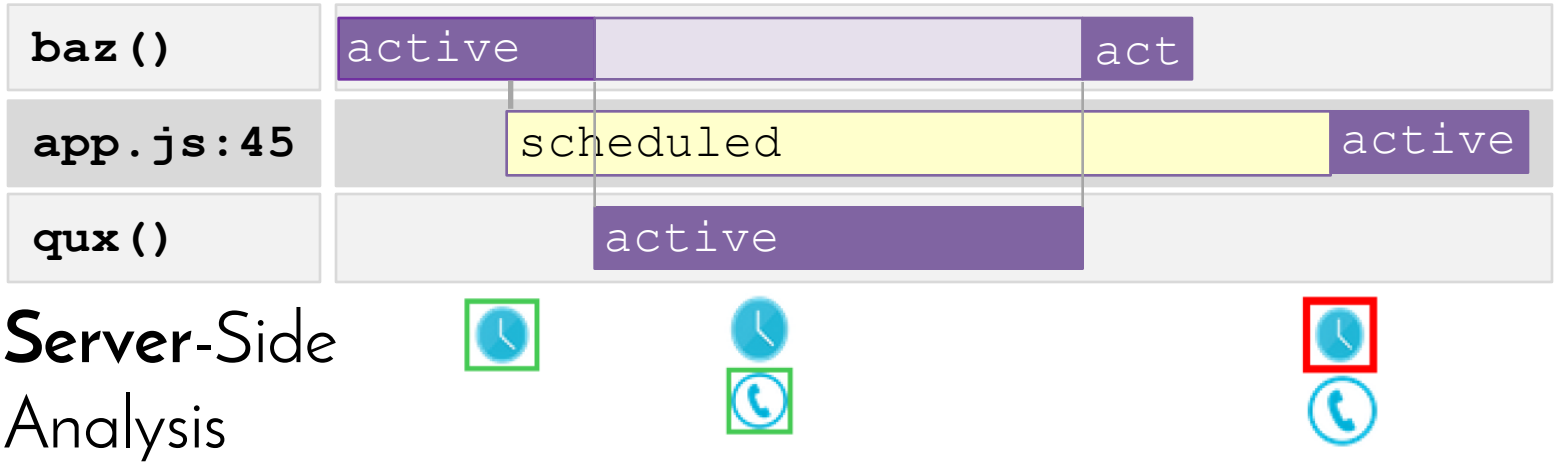



**Server**-Side Analysis

# Visualization

**Client**-Side Analysis


Time ⟶ Structure of time ⟶ Linear & Branching

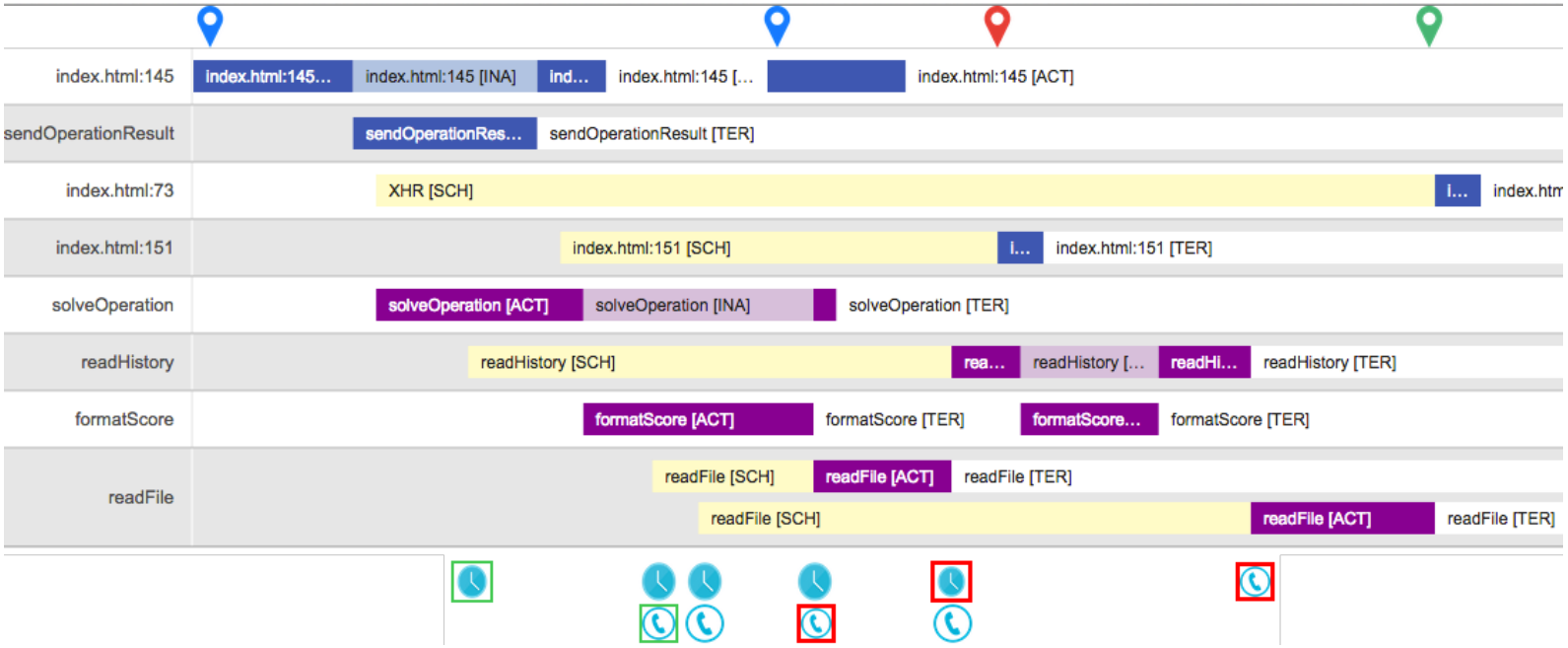



**Server**-Side Analysis

# Implementation: *Sahand*

- Express.js application
- Proxy -> dynamic instrumentation
- Esprima, Estraverse, Escodegen

https://github.com/saltlab/sahand



# Evaluation

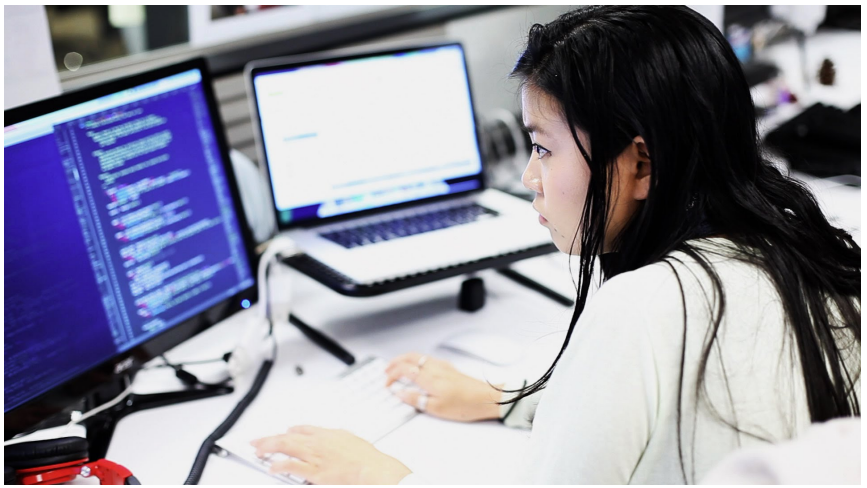
Does using ***Sahand*** improve developers' performance in program comprehension tasks?

# Controlled Experiment

- ***Sahand***'s effect on developers' performance
- 12 Participants
- Object: full-stack JavaScript application

# Controlled Experiment

- Design
  - Control: tool and expertise level
  - Measure: performance
- Procedure
  - Pre-questionnaire
  - Tutorial
  - Tasks
  - Post-questionnaire

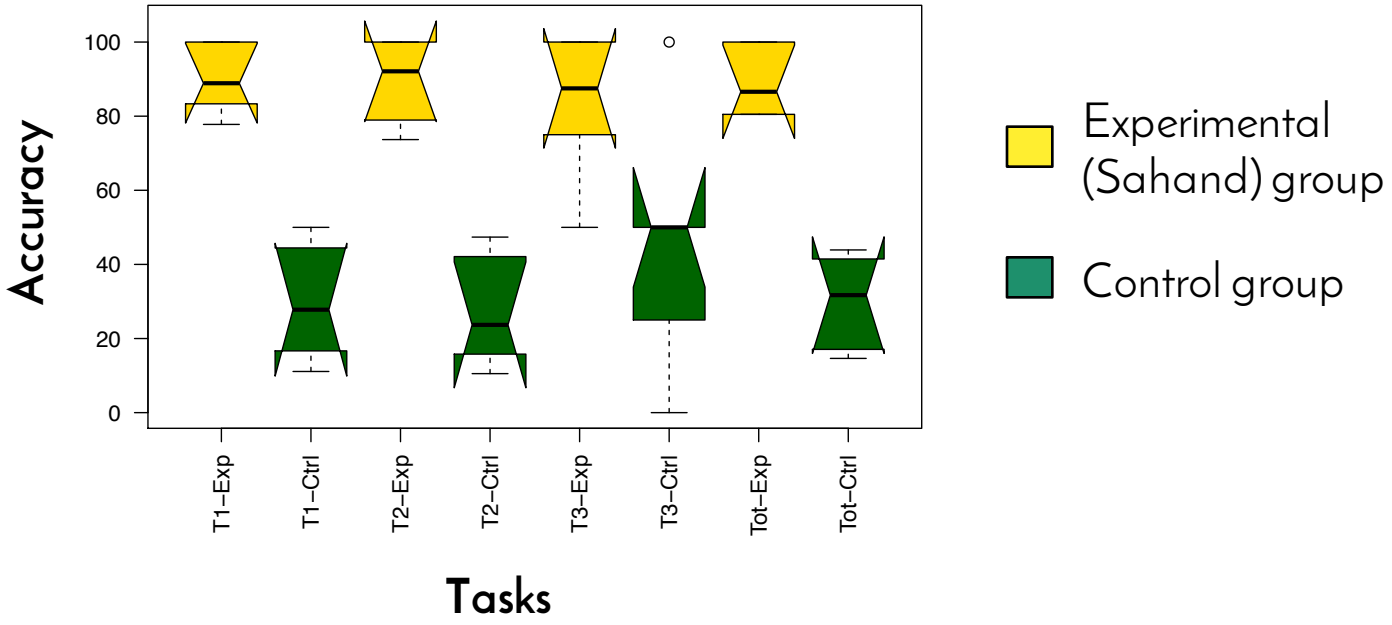


# Results Highlight

Using ***Sahand***

**3 times more accuracy**

In the same time



# *Sahand*: http://github.com/saltlab/sahand

## Summary of Challenges

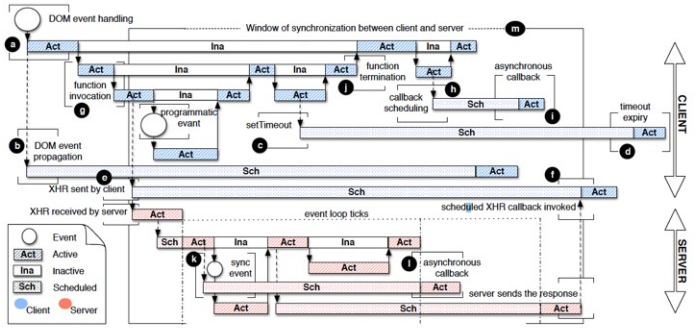
- Server-side callbacks
- Network communication
- Asynchronous client side

Related work:

| Zaidman et al. | Hibschman et al. | Alimadadi et al. |
|---|---|---|
| EMSE'13 | UIST'14 | ICSE'14, ECOOP'15 |

6

## Behavioral Model: Example

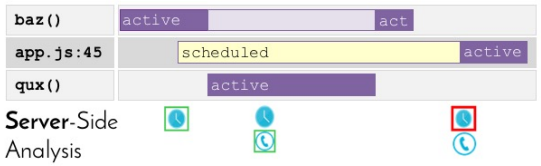



11

## Visualization

**Client**-Side Analysis


Connecting client and server




**Server**-Side Analysis

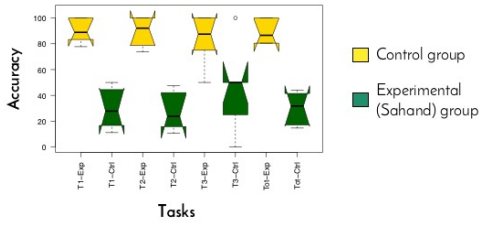
12

## Results Highlight

Using Sahand

**3 times more accuracy**

In the same time




20

Saba Alimadadi

**Hire Me!**