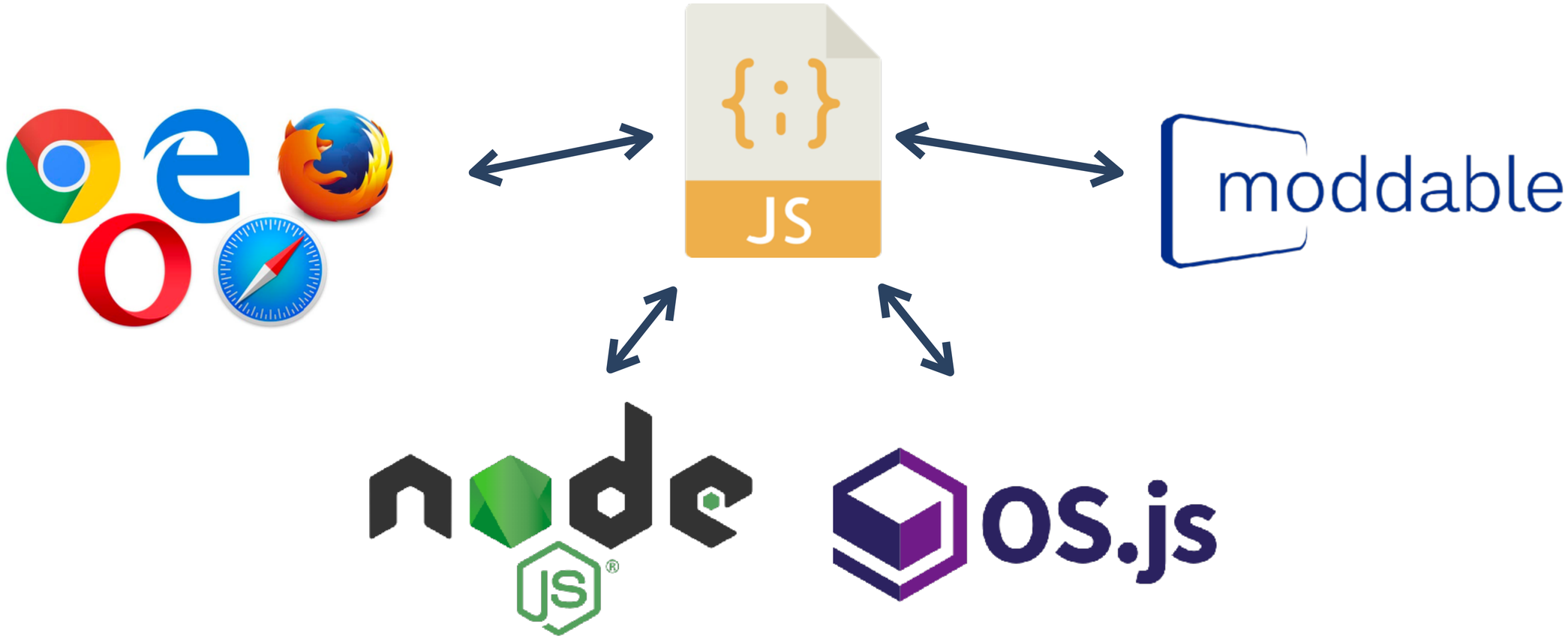
JSTAR: JavaScript Specification Type Analyzer using Refinement

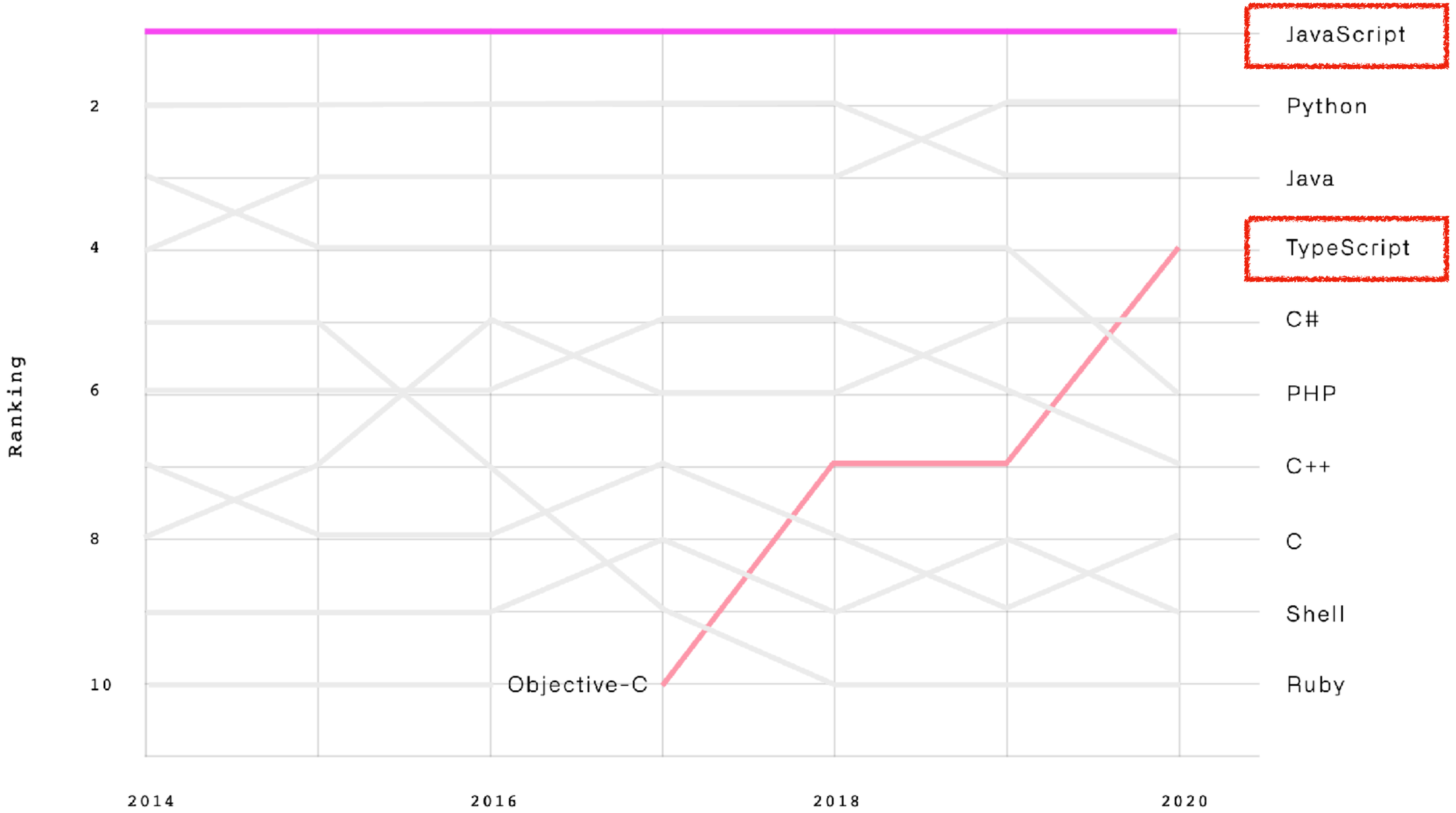
The 36th IEEE/ACM International Conference on Automated Software Engineering (ASE'21)

Jihyeok Park, Seungmin An, Wonho Shin,
Yusung Sim, Sukyoung Ryu

PLRG @ KAIST
November 17, 2021

JavaScript is Everywhere





https://octoverse.github.com/

ECMAScript: JavaScript Specification

Syntax

ArrayLiteral[Yield, Await] :

[*Elision*opt]

[*ElementList*[?Yield, ?Await]]

[*ElementList*[?Yield, ?Await] , *Elision*opt]

The production of *ArrayLiteral* in ES12

Semantics

13.2.5.2 Runtime Semantics: Evaluation

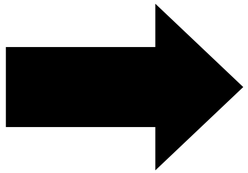
ArrayLiteral : [*ElementList* , *Elision*opt]

1. Let *array* be ! ArrayCreate(0).
2. Let *nextIndex* be the result of performing ArrayAccumulation for *ElementList* with arguments *array* and 0.
3. ReturnIfAbrupt(*nextIndex*).
4. If *Elision* is present, then
   a. Let *len* be the result of performing ArrayAccumulation for *Elision* with arguments *array* and *nextIndex*.
   b. ReturnIfAbrupt(*len*).
5. Return *array*.

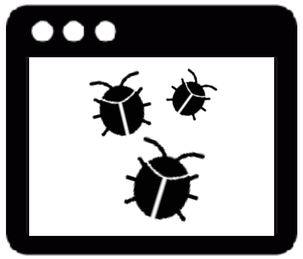
The Evaluation algorithm for the third alternative of *ArrayLiteral* in ES12

Correctness of ECMAScript is Important

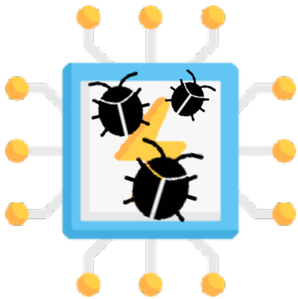
ECMAScript



Web Applications

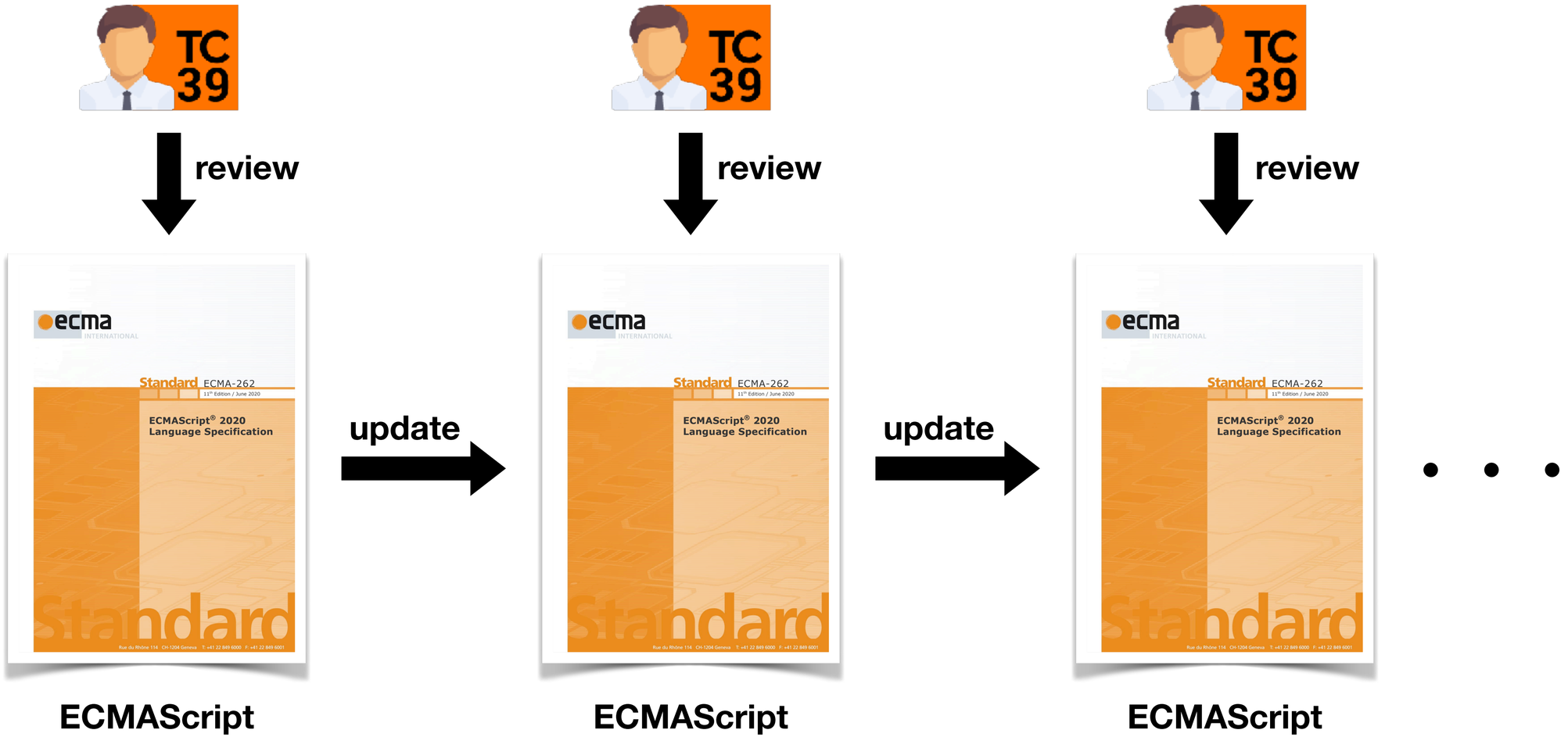


Server-side Programs

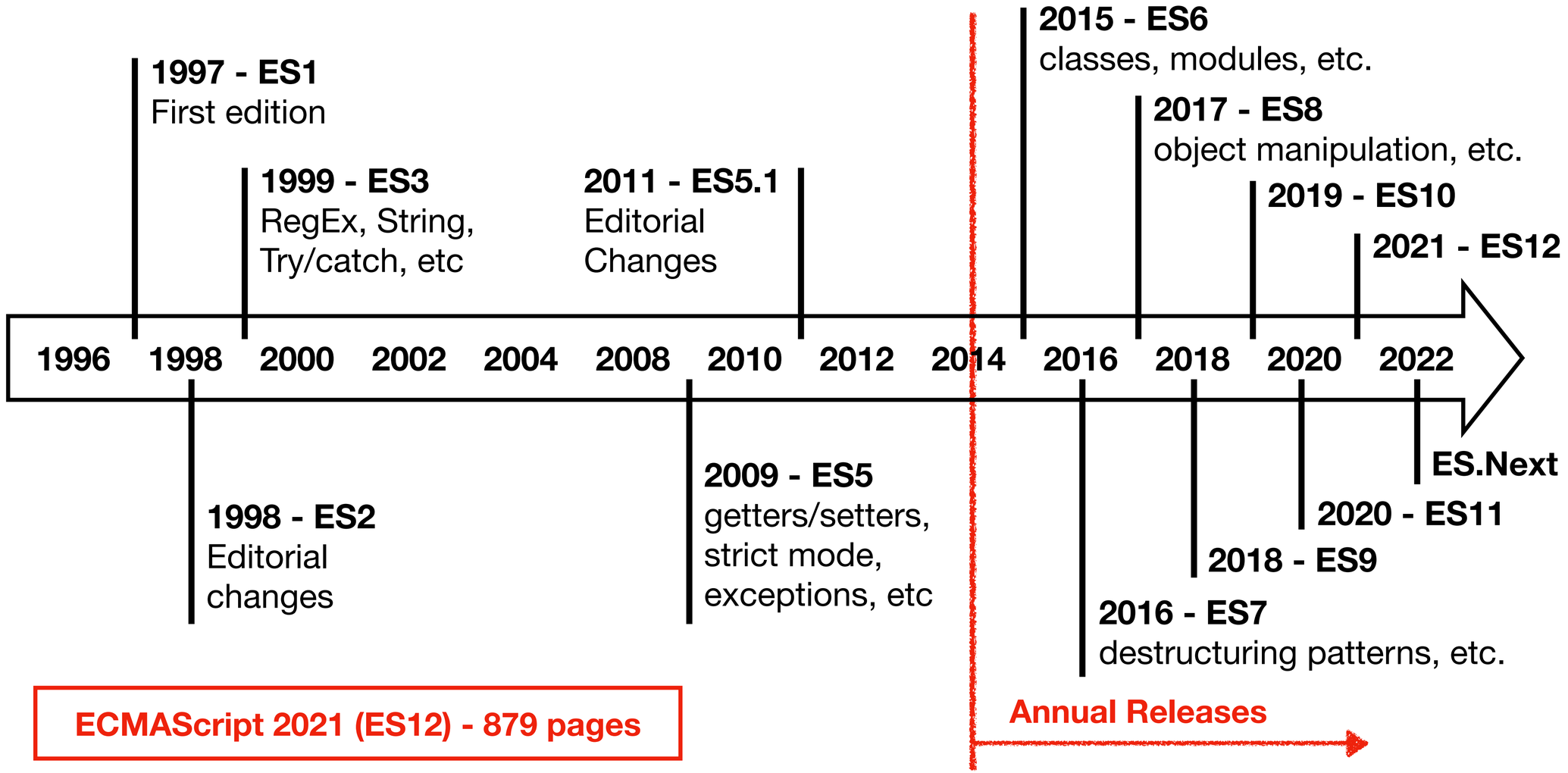


Embedded Systems

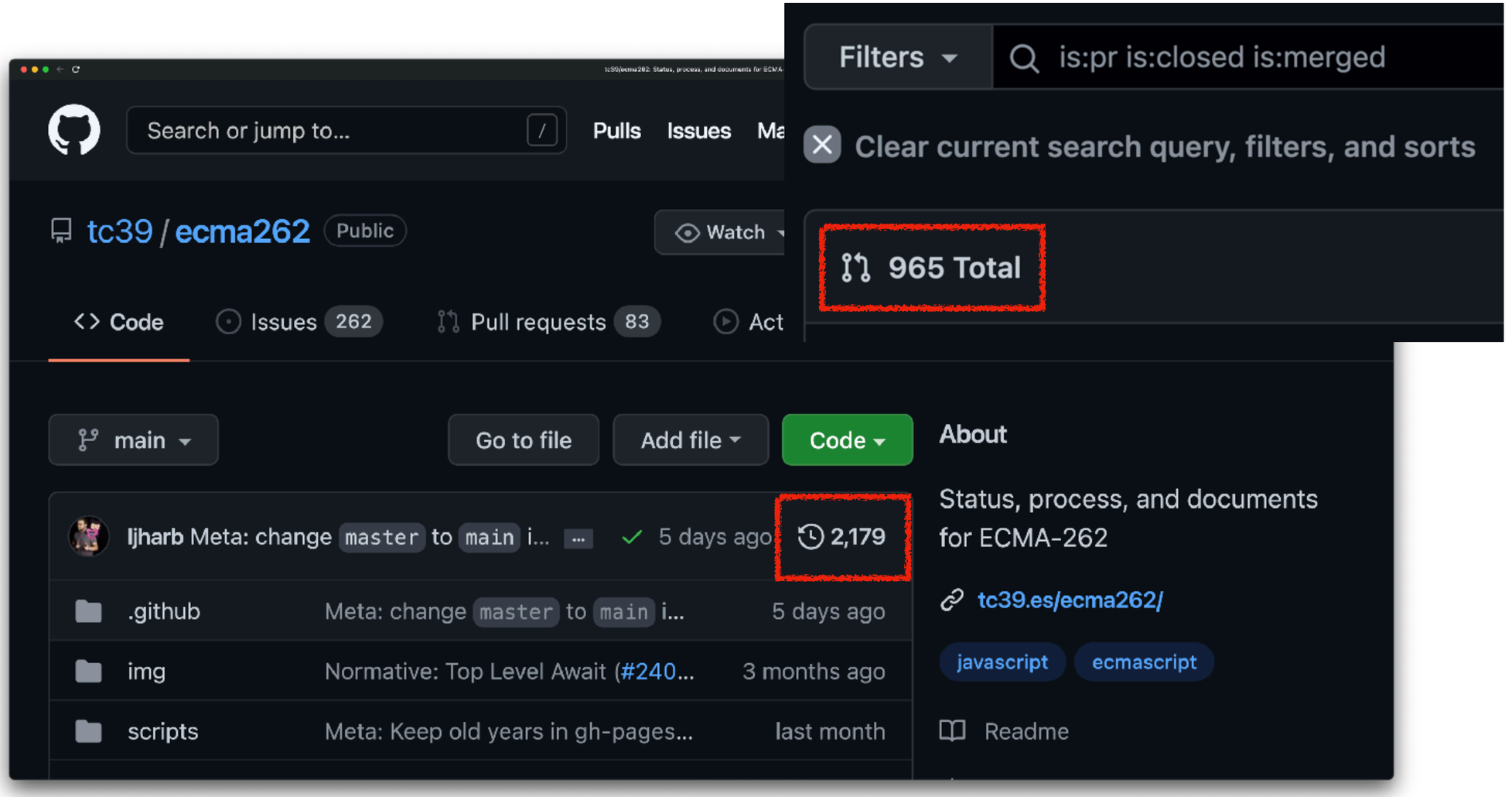
Problem: Manual Review of ECMAScript



Problem: Fast Evolving JavaScript



Problem: Open Development Process



Solution: Type Analysis for ECMAScript

20.3.2.28 Math.round (*x*) `x: (String ∨ Boolean ∨ Number ∨ Object ∨ ...)`

1. Let *n* be ? ToNumber(*x*). `n: (Number) ∧ ToNumber(x): (Number ∨ Exception)`
2. If *n* is an integral Number, return *n*.
3. If $x < 0.5$ and $x > 0$, return **+0**.
4. If $x < 0$ and $x \geq -0.5$, return **-0**.

...

Type Mismatch for numeric operator `>`

```
Math.round(true)  = ???
Math.round(false) = ???
```

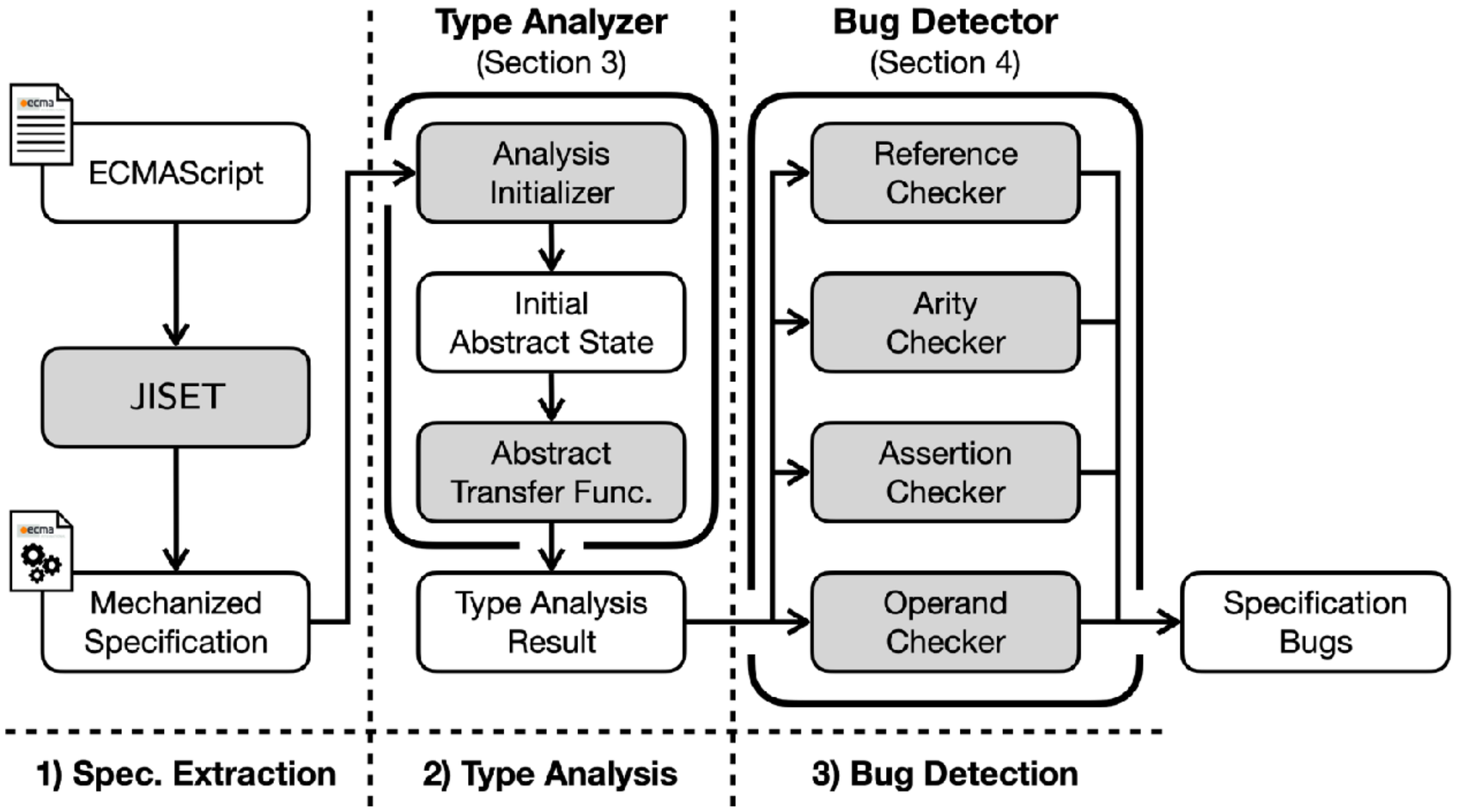
3. If $n < 0.5$ and $n > 0$, return **+0**.
4. If $n < 0$ and $n \geq -0.5$, return **-0**.

```
Math.round(true)  = 1
Math.round(false) = 0
```

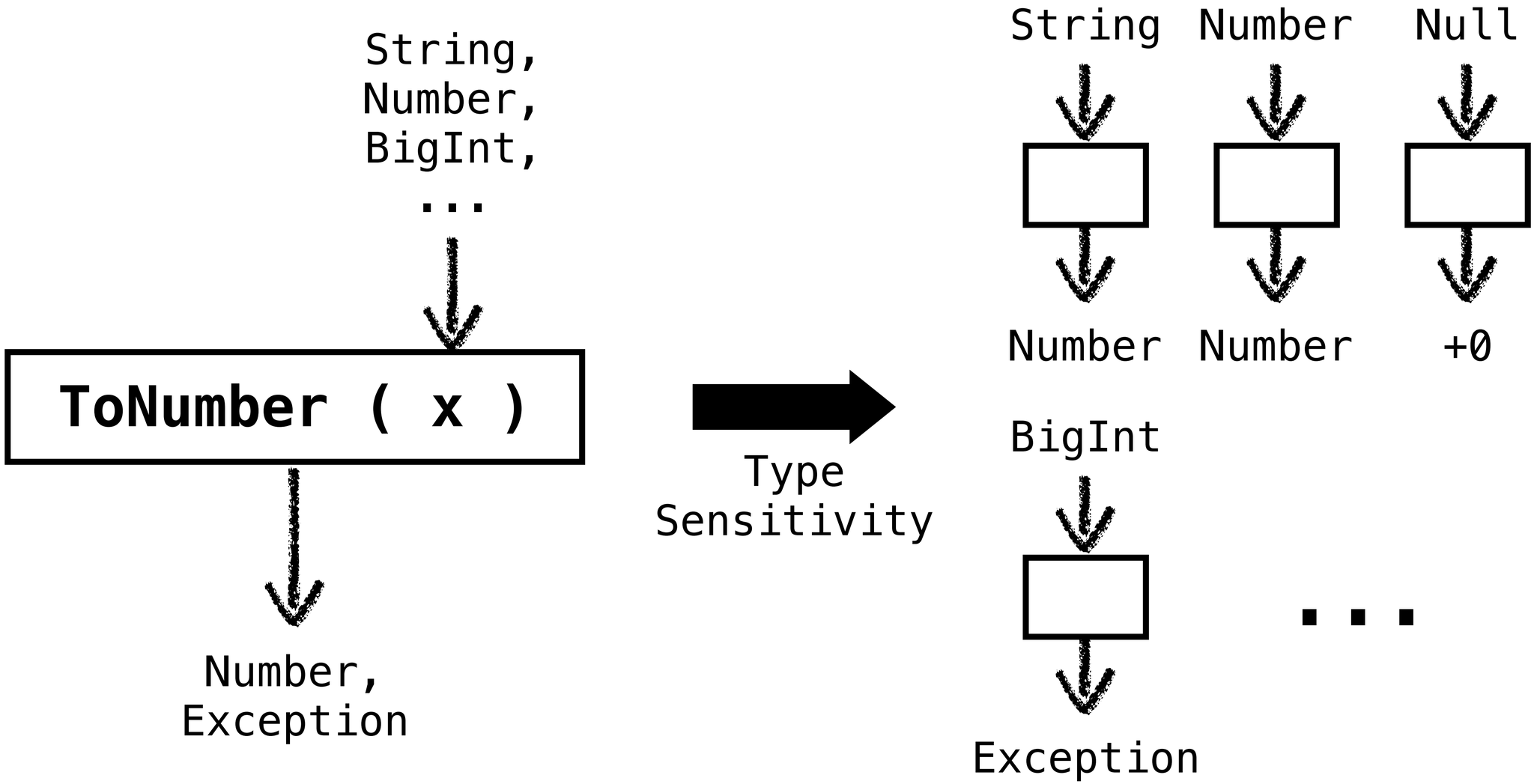
https://github.com/tc39/ecma262/tree/575149cfd77aebcf3a129e165bd89e14caafc31c

Overall Structure of JSTAR

JavaScript Specification Type Analyzer using Refinement



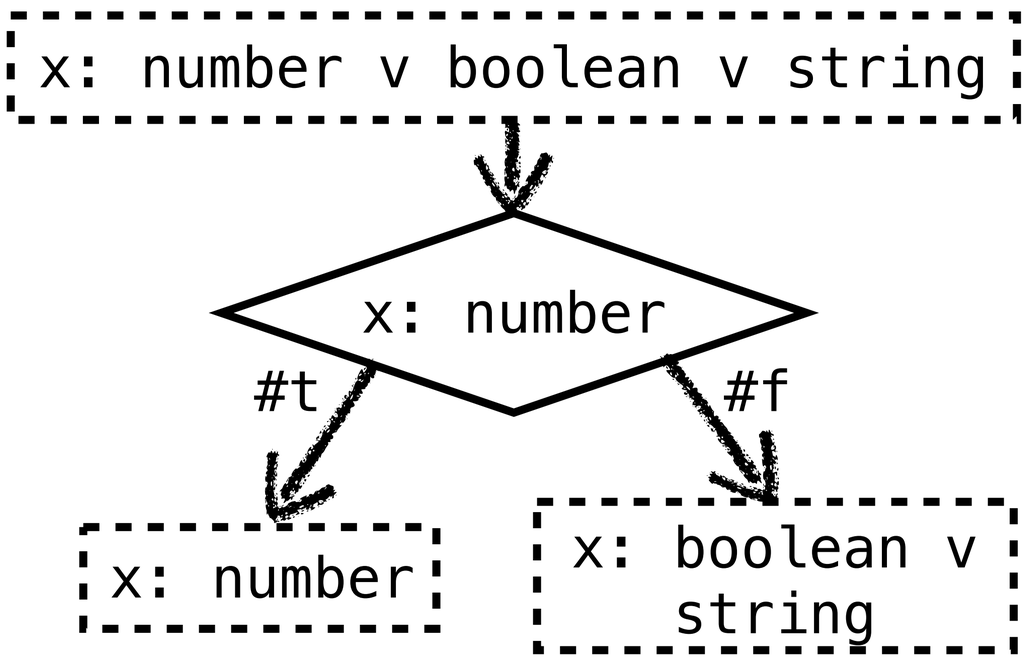
Precision ⇧ - 1) Type Analysis



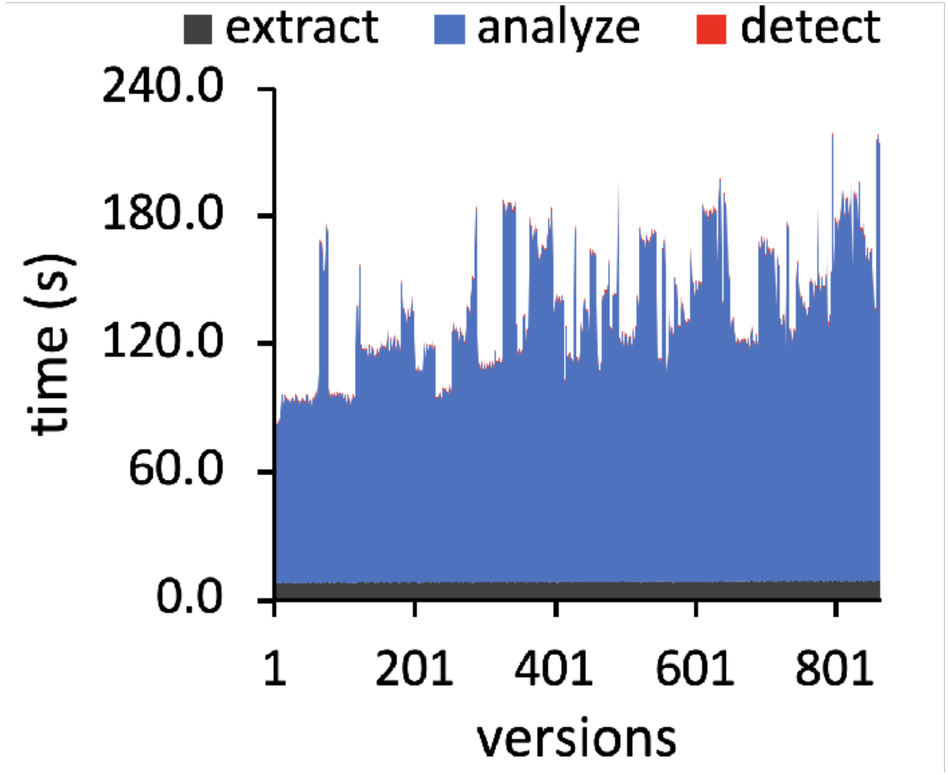
Precision ⇧ - 2) Condition-based Refinement

$$
\begin{aligned}
\texttt{refine}(!e, b)(\sigma^\sharp) &= \texttt{refine}(e, \neg b)(\sigma^\sharp) \\
\texttt{refine}(e_0 \texttt{ || } e_1, b)(\sigma^\sharp) &= \begin{cases} \sigma_0^\sharp \sqcup \sigma_1^\sharp & \text{if } b \\ \sigma_0^\sharp \sqcap \sigma_1^\sharp & \text{if } \neg b \end{cases} \\
\texttt{refine}(e_0 \texttt{ \&\& } e_1, b)(\sigma^\sharp) &= \begin{cases} \sigma_0^\sharp \sqcap \sigma_1^\sharp & \text{if } b \\ \sigma_0^\sharp \sqcup \sigma_1^\sharp & \text{if } \neg b \end{cases} \\
\texttt{refine}(\texttt{x.Type} == c_{\texttt{normal}}, \texttt{\#t})(\sigma^\sharp) &= \sigma^\sharp[\texttt{x} \mapsto \tau_{\texttt{x}}^\sharp \sqcap \texttt{normal}(\mathbb{T})] \\
\texttt{refine}(\texttt{x.Type} == c_{\texttt{normal}}, \texttt{\#f})(\sigma^\sharp) &= \sigma^\sharp[\texttt{x} \mapsto \tau_{\texttt{x}}^\sharp \sqcap \{\texttt{abrupt}\}] \\
\texttt{refine}(\texttt{x} == e, \texttt{\#t})(\sigma^\sharp) &= \sigma^\sharp[\texttt{x} \mapsto \tau_{\texttt{x}}^\sharp \sqcap \tau_e^\sharp] \\
\texttt{refine}(\texttt{x} == e, \texttt{\#f})(\sigma^\sharp) &= \sigma^\sharp[\texttt{x} \mapsto \tau_{\texttt{x}}^\sharp \setminus \lfloor \tau_e^\sharp \rfloor] \\
\texttt{refine}(\texttt{x} : \tau, \texttt{\#t})(\sigma^\sharp) &= \sigma^\sharp[\texttt{x} \mapsto \tau_{\texttt{x}}^\sharp \sqcap \{\tau\}] \\
\texttt{refine}(\texttt{x} : \tau, \texttt{\#f})(\sigma^\sharp) &= \sigma^\sharp[\texttt{x} \mapsto \tau_{\texttt{x}}^\sharp \setminus \{\tau' \mid \tau' <: \tau\}] \\
\texttt{refine}(e, b)(\sigma^\sharp) &= \sigma^\sharp
\end{aligned}
$$

where $\sigma_j^\sharp = \texttt{refine}(e_j, b)(\sigma^\sharp)$ for $j = 0, 1$, $\tau_e^\sharp = [\![e]\!]_e^\sharp(\sigma^\sharp)$, and $\lfloor \tau^\sharp \rfloor$ returns $\{\tau\}$ if $\tau^\sharp$ denotes a singleton type τ, or returns $\varnothing$, otherwise.



RQ1) Performance

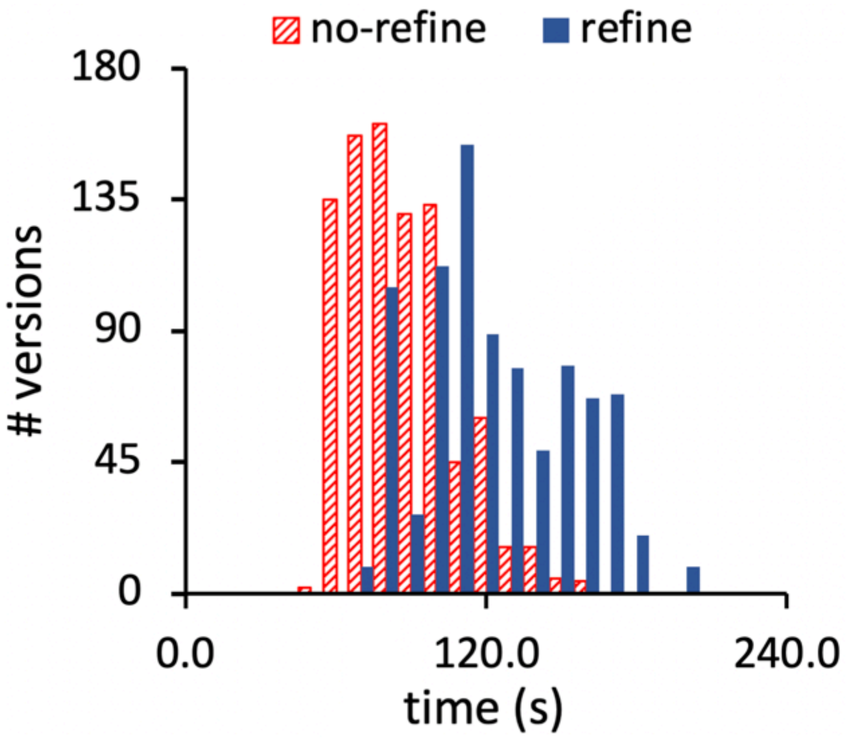


- **864 versions** of ECMAScript (Jan. 1, 2018 to Mar. 9, 2021)
- 4.2GHz Quad-Core Intel Core i7
- 32GB of RAM
- **Average Time : 137.3 s**
  - extract : 8.0 s
  - analyze: 128.5
  - detect: 0.8 s

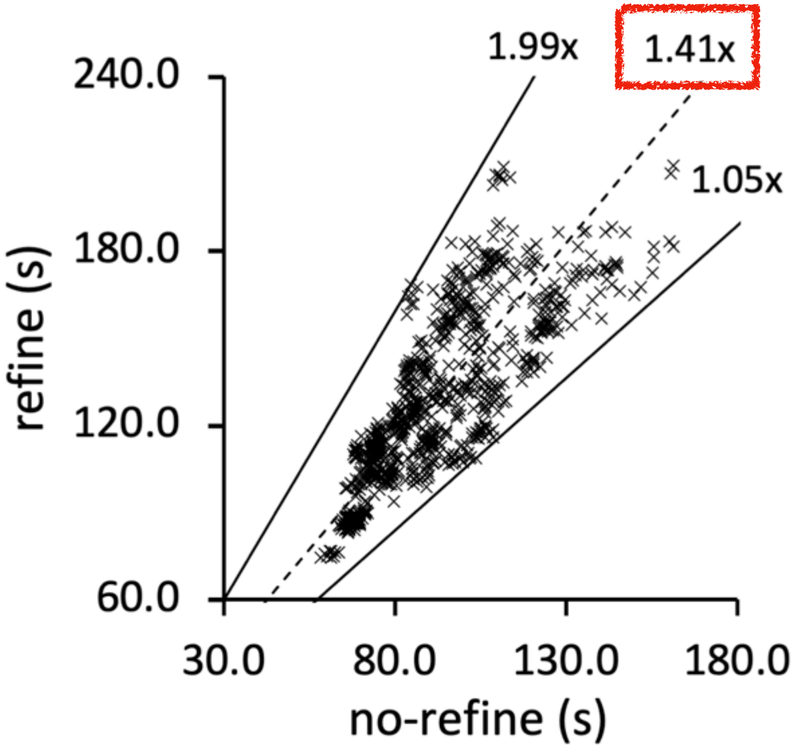
RQ2) Precision

Checker	Bug Kind	Precision = (# True Bugs) / (# Detected Bugs)					
		no-refine		refine		Δ	
Reference	UnknownVar	62 / 106	17 / 60	63 / 78	17 / 31	+1 / -28	/ -29
	DuplicatedVar		45 / 46		46 / 47		+1 / +1
Arity	MissingParam	4 / 4	4 / 4	4 / 4	4 / 4	/	/
Assertion	Assertion	4 / 56	4 / 56	4 / 31	4 / 31	/ -25	/ -25
Operand	NoNumber	22 / 113	2 / 65	22 / 44	2 / 6	/ -69	/ -59
	Abrupt		20 / 48		20 / 38		/ -10
Total		92 / 279 (33.0%)		93 / 157 (59.2%)		+1 / -122 (+26.3%)	

RQ3) Effectiveness of Refinement



(c) The histogram of time



(d) The ratio of time

)etected Bugs)

Δ	
+1 / -28	/ -29
	+1 / +1
/	/
/ -25	/ -25
/ -69	/ -59
	/ -10
+1 / -122	(+26.3%)

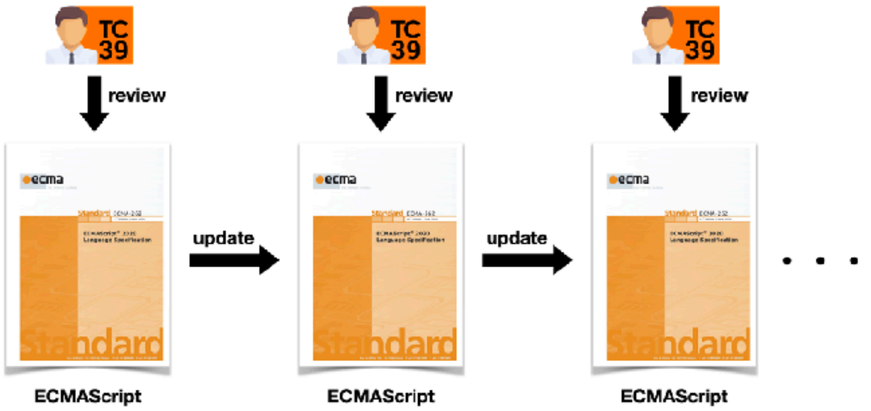
RQ4) Detection of New Bugs

- The Latest Version: **ECMAScript 2021 (ES12)**

14 Bugs in Spec.

Name	Feature	#	Checker	Created	Life Span
ES12-1	Switch	3	Reference	2015-09-22	1,996 days
ES12-2	Try	3	Reference	2015-09-22	1,996 days
ES12-3	Arguments	1	Reference	2015-09-22	1,996 days
ES12-4	Array	2	Reference	2015-09-22	1,996 days
ES12-5	Async	1	Reference	2015-09-22	1,996 days
ES12-6	Class	1	Reference	2015-09-22	1,996 days
ES12-7	Branch	1	Reference	2015-09-22	1,996 days
ES12-8	Arguments	2	Operand	2015-12-16	1,910 days

Problem: Manual Review of ECMAScript



Solution: Type Analysis for ECMAScript

20.3.2.28 Math.round (*x*) `x: (String ∨ Boolean ∨ Number ∨ Object ∨ ...)`

1. Let *n* be ? ToNumber(*x*). `n: (Number) ∧ ToNumber(x): (Number ∨ Exception)`
2. If *n* is an integral Number, return *n*.
3. If $x < 0.5$ and $x > 0$, return +0.
4. If $x < 0$ and $x \geq -0.5$, return -0.

...

Type Mismatch for numeric operator `>`

```
Math.round(true)  = ???
Math.round(false) = ???
```

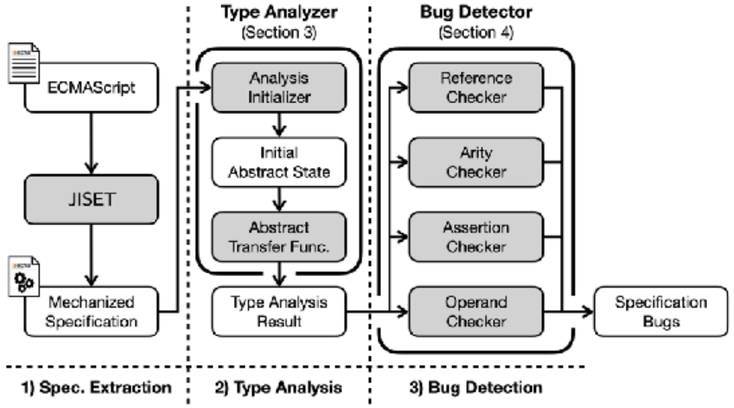
3. If $n < 0.5$ and $n > 0$, return +0.
4. If $n < 0$ and $n \geq -0.5$, return -0.

```
Math.round(true)  = 1
Math.round(false) = 0
```

https://github.com/tc39/ecma262/tree/575149cfd77aebcf3a129e165bd89e14caafc31c

Overall Structure of JSTAR

JavaScript Specification Type Analyzer using Refinement



RQ4) Detection of New Bugs

- The Latest Version: **ECMAScript 2021 (ES12)**

14 Bugs in Spec.

Name	Feature	#	Checker	Created	Life Span
ES12-1	Switch	3	Reference	2015-09-22	1,996 days
ES12-2	Try	3	Reference	2015-09-22	1,996 days
ES12-3	Arguments	1	Reference	2015-09-22	1,996 days
ES12-4	Array	2	Reference	2015-09-22	1,996 days
ES12-5	Async	1	Reference	2015-09-22	1,996 days
ES12-6	Class	1	Reference	2015-09-22	1,996 days
ES12-7	Branch	1	Reference	2015-09-22	1,996 days
ES12-8	Arguments	2	Operand	2015-12-16	1,910 days