Lightweight threads, actors, async/await, ..., in a minimal extension of the WebAssembly reference interpreter

Sam Lindley

The University of Edinburgh

5th April 2021 and 19th April 2021

Wasm effect handler proposal

(Daniel Hillerström, Daan Leijen, Sam Lindley, Matija Pretnar, Andreas Rossberg, KC Sivamarakrishnan)

Typed continuations to model stacks
https://github.com/WebAssembly/design/issues/1359

Formal spec
https://github.com/effect-handlers/wasm-spec/blob/master/proposals/continuations/Overview.md

Reference interpreter extension
https://github.com/effect-handlers/wasm-spec

Examples
https://github.com/effect-handlers/wasm-spec/tree/master/proposals/continuations/examples

Events

Synonyms: operation, command, resumable exception

event $\$e$ (**param** *s1*)$*$ (**result** *s2*)$*$	declare event of type $[s1*] \rightarrow [s2*]$
suspend $\$e : [s1*] \rightarrow [s2*]$	invoke event
where e is an event of type $[s1*] \rightarrow [s2*]$	

Continuations

Synonyms: stacklet, resumption

cont.new : $[(\textbf{ref}\ \$ft)] \rightarrow [(\textbf{cont}\ \$ft)]$ — new continuation from function

where $\$ft$ denotes a function type $[t1*] \rightarrow [t2*]$

resume $(\textbf{event}\ \$e\ \$l)* : [t1*\ (\textbf{cont}\ \$ft)] \rightarrow [t2*]$ — invoke continuation with handler

where $\$ft$ denotes a function type $[t1*] \rightarrow [t2*]$

each $\$e$ is an event and

each $\$l$ is a label pointing to its handler clause

if $\$e : [s1*] \rightarrow [s2*]$ then

$\$l : [s1*\ (\textbf{cont}\ \$k)] \rightarrow [t2*]$

$\$k : [s2*] \rightarrow [t2*]$

Continuations

Synonyms: stacklet, resumption

cont.new : $[(\textbf{ref}\ \$ft)] \to [(\textbf{cont}\ \$ft)]$ — new continuation from function

where $\$ft$ denotes a function type $[t1*] \to [t2*]$

resume $(\textbf{event}\ \$e\ \$l)* : [t1*\ (\textbf{cont}\ \$ft)] \to [t2*]$ — invoke continuation with handler

where $\$ft$ denotes a function type $[t1*] \to [t2*]$

each $\$e$ is an event and

each $\$l$ is a label pointing to its handler clause

if $\$e : [s1*] \to [s2*]$ then

$\$l : [s1*\ (\textbf{cont}\ \$k)] \to [t2*]$

$\$k : [s2*] \to [t2*]$

resume_throw $\$exn : [te*\ (\textbf{cont}\ \$ft)] \to [t2*]$ — discard cont. and throw exception

where $\$ft$ denotes a function type $[t1*] \to [t2*]$

$\$exn : [te*] \to []$

Encoding handlers with blocks and labels

If $e1 : [s1*] \to [t1*], \ldots, en : [sn*] \to [tn*]$ then a typical handler looks something like:

```
(loop $l
  (block $on_e1 (result [s1* (cont ([t1*] → [tr*]))])
    ...
    (block $on_en (result [sn* (cont ([tn*] → [tr*]))])
      (resume
        (event $e1 $on_e1) ... (event $en $on_en)
        (local.get $nextk))
      ...
    ) ;;     $on_en (result [sn* (cont ([tn*] → [tr*]))])
    bn
    (br $l)
  ) ;;     $on_e1 (result [s1* (cont ([t1*] → [tr*]))])
  b1
  (br $l))
```

- Structured as a scheduler loop
- Handler body comes *after* block
- Result specifies types of parameters and continuation

Dependencies

Function references

Exceptions

Not GC

Examples

Lightweight threads

Actors

Async/await

...

https://github.com/effect-handlers/wasm-spec/tree/examples/proposals/continuations/examples

Partial continuation application

Analogous to **func.bind** in the function references proposal — but lifetime is predictable as continuations are one-shot

$$\textbf{cont.bind}\ \$ct : [t3* \ (\textbf{ref}\ \$ct')] \to [(\textbf{ref}\ \$ct)]$$
$$\text{where}\ \$ct = \textbf{cont}\ \$ft$$
$$\$ft = [t1*] \to [t2*]$$
$$\$ct' = \textbf{cont}\ \$ft'$$
$$\$ft' = [t3*\ t1*] \to [t2*]$$

Partial continuation application

Analogous to **func.bind** in the function references proposal — but lifetime is predictable as continuations are one-shot

$$\begin{aligned}
&\textbf{cont.bind}\ \$ct : [t3* \ (\textbf{ref}\ \$ct')] \to [(\textbf{ref}\ \$ct)] \\
&\quad \text{where}\ \$ct = \textbf{cont}\ \$ft \\
&\qquad\qquad \$ft = [t1*] \to [t2*] \\
&\qquad\qquad \$ct' = \textbf{cont}\ \$ft' \\
&\qquad\qquad \$ft' = [t3*\ t1*] \to [t2*]
\end{aligned}$$

Avoids code duplication

Partial continuation application

Analogous to **func.bind** in the function references proposal — but lifetime is predictable as continuations are one-shot

$$\begin{aligned}&\textbf{cont.bind}\ \$ct : [t3*\ (\textbf{ref}\ \$ct')] \to [(\textbf{ref}\ \$ct)]\\ &\quad \text{where } \$ct = \textbf{cont}\ \$ft\\ &\qquad \$ft = [t1*] \to [t2*]\\ &\qquad \$ct' = \textbf{cont}\ \$ft'\\ &\qquad \$ft' = [t3*\ t1*] \to [t2*]\end{aligned}$$

Avoids code duplication

Should we simplify **resume**?

$$\textbf{resume}\ (\textbf{event}\ \$e\ \$l)* : [t1*\ (\textbf{cont}\ \$ft)] \to [t2*] \text{ where } \$ft = [t1*] \to [t2*]$$

versus

$$\textbf{resume}\ (\textbf{event}\ \$e\ \$l)* : [(\textbf{cont}\ \$ft)] \to [t2*] \text{ where } \$ft = [] \to [t2*]$$

Links

Typed continuations to model stacks
https://github.com/WebAssembly/design/issues/1359

Formal spec
https://github.com/effect-handlers/wasm-spec/blob/master/proposals/continuations/Overview.md

Reference interpreter extension
https://github.com/effect-handlers/wasm-spec

Examples
https://github.com/effect-handlers/wasm-spec/tree/master/proposals/continuations/examples