TopHat: a stylish journey through modular interactive workflows

Tim Steenvoorden
Markus Klinik
Nico Naus

Dutch FP Day, January 11, 2019

Institute for Computing and Information Sciences
Radboud University

Task Oriented Programming (TOP)

Workflows
coordinate collaboration

Interactive
driven by user input

Modular
higher order

elementary building blocks and concepts

labeled transition system

embedding in simply typed λ-calculus
(with clearly separated semantics)

FOUNDATION FOR FORMAL REASONING
AND COMPARISON TO OTHER FRAMEWORKS

Institute for Computing
and Information Sciences
Radboud University

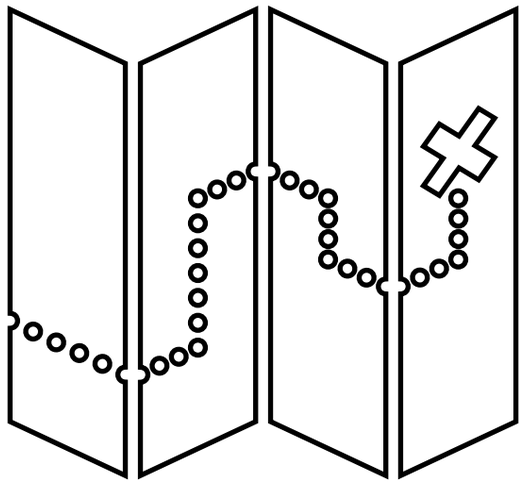
Building blocks

Institute for Computing
and Information Sciences
Radboud University

Editors

□5

Empty → ⊠Int

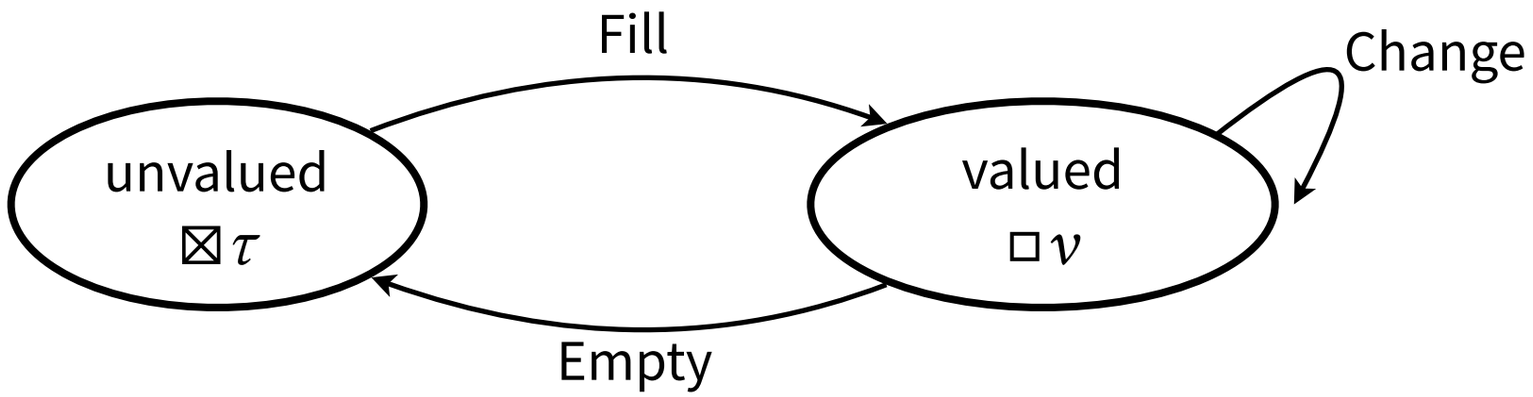


□(51°49'N, 5°52'E)

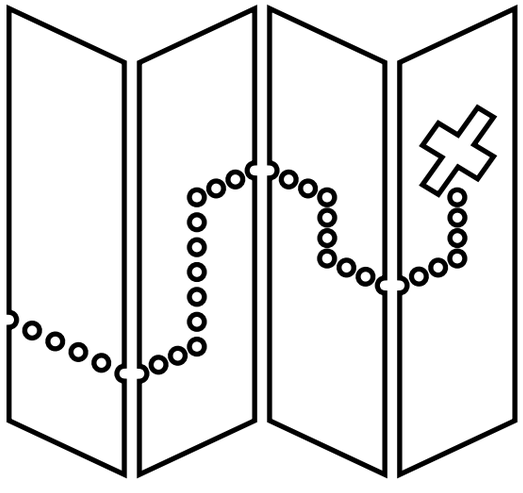
□True

False → □False

CAPTURE THE EVER CHANGING NATURE OF WIDGETS

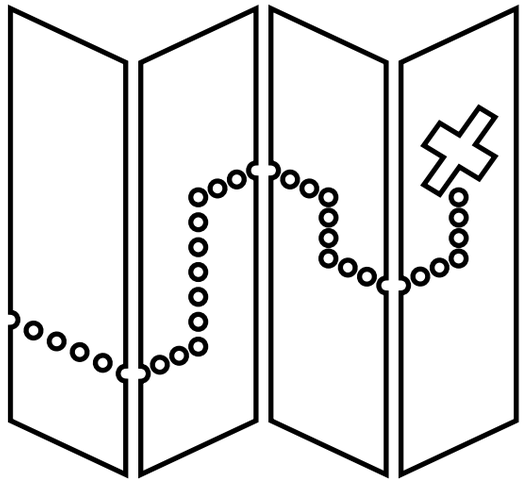


Shared editors

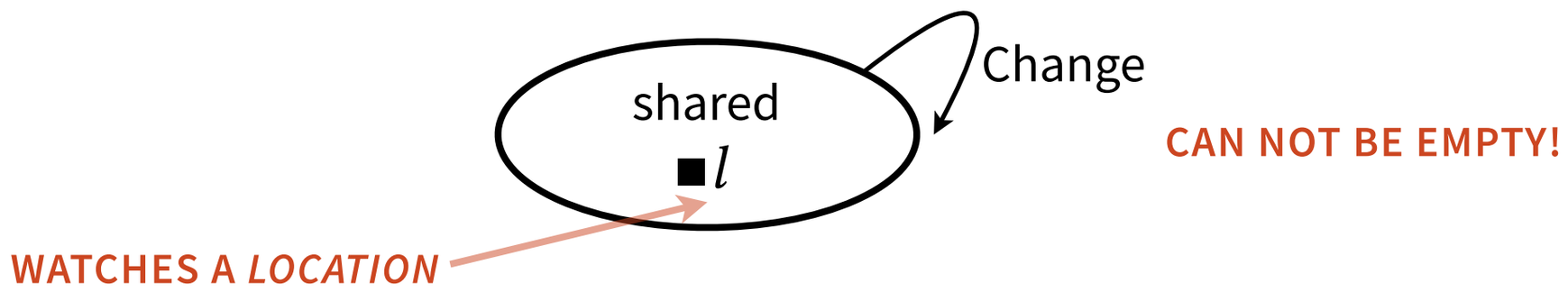


■(51°49'N, 5°52'E)

(51°49'N, 5°52'W) → ■(51°49'N, 5°52'W)

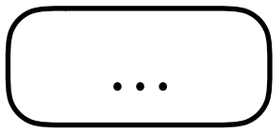


■(51°49'N, 5°52'E)

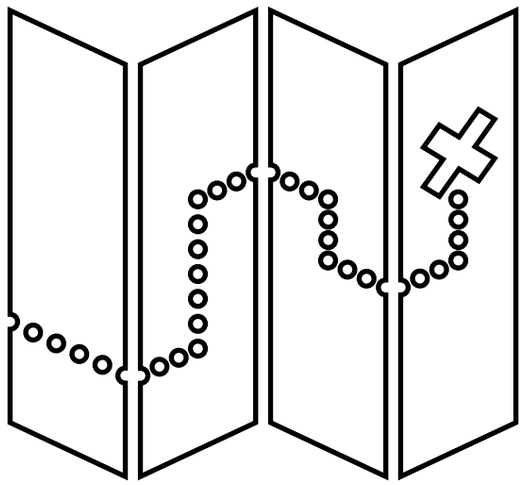


CAN NOT BE EMPTY!

Observations



$\mathcal{V}(\boxtimes \text{INT}) = \bot$



$\mathcal{V}(\square(51°49'\text{N}, 5°52'\text{E}))$
$= (51°49'\text{N}, 5°52'\text{E})$

$\mathcal{V}(\square \text{True}) = \text{True}$

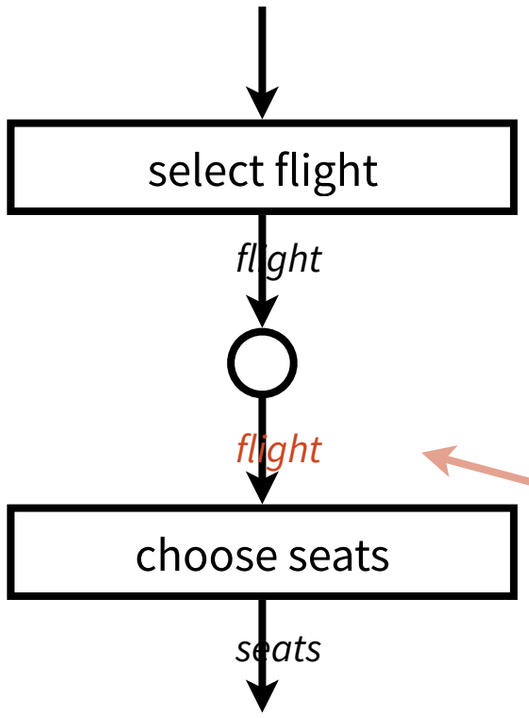
SOME TASKS DO NOT HAVE A VALUE

value $\quad \mathcal{V} : \text{TASK } \tau \longrightarrow \text{MAYBE } \tau$

user interface $\quad \mathcal{U} : \text{TASK } \tau \longrightarrow \text{HTML}$ or $\mathcal{U} : \text{TASK } \tau \longrightarrow \text{STRING}$ or $\mathcal{U} : \text{TASK } \tau \longrightarrow \ldots$

possible inputs $\quad \mathcal{J} : \text{TASK } \tau \longrightarrow \text{LIST INPUT}$

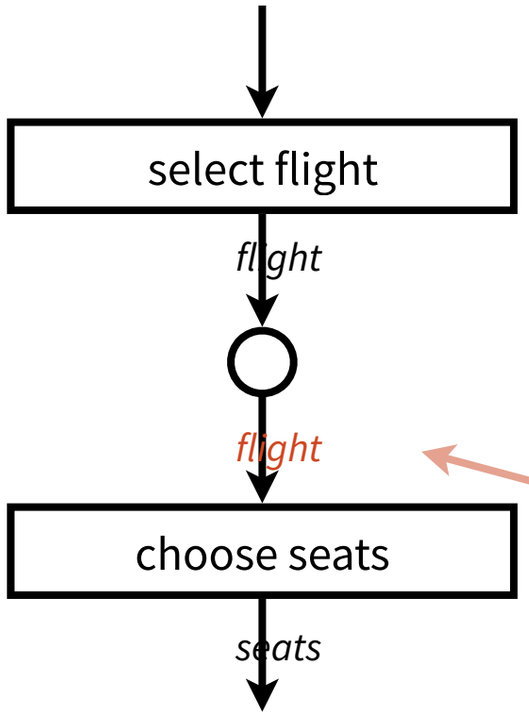
Steps



select_flight ▸ choose_seats

USE VALUE IN NEXT TASK?

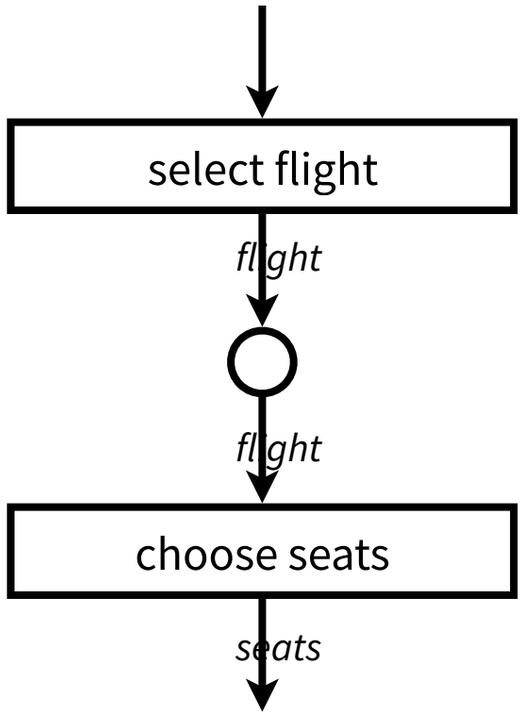
Steps



$$\text{select\_flight} \blacktriangleright \lambda \mathit{flight}.\ \text{choose\_seats}\ \mathit{flight}$$

USE VALUE IN NEXT TASK?

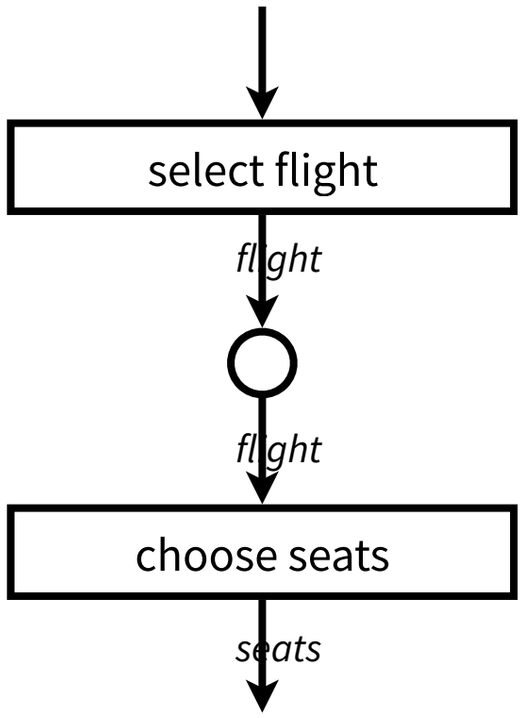
Steps



select_flight ▸ λ*flight*. choose_seats *flight*

WHEN TO PROCEED TO THE NEXT TASK?

Steps

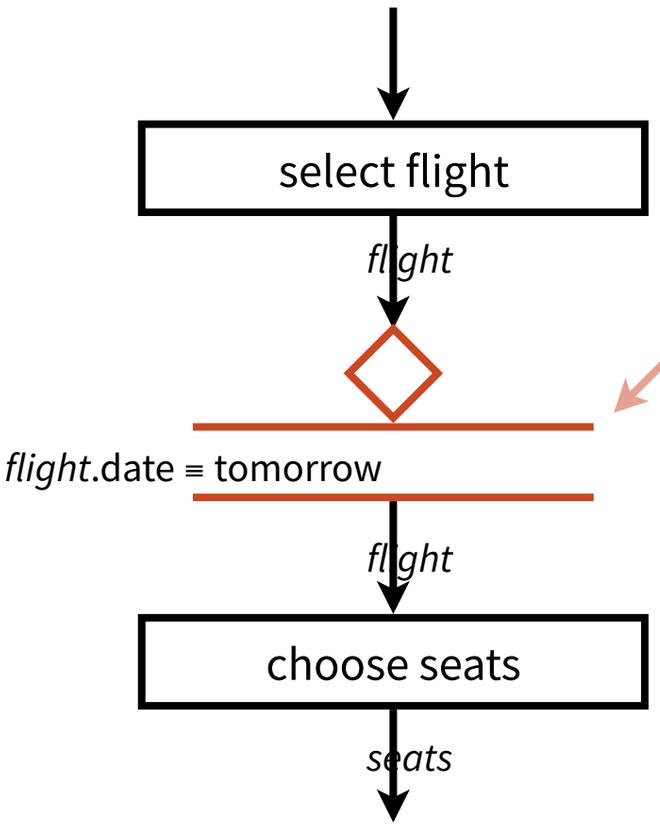


$$\text{select\_flight} \blacktriangleright \lambda \mathit{flight}.\ \text{choose\_seats}\ \mathit{flight}$$

WHEN TO PROCEED TO THE NEXT TASK?

$$\Longrightarrow \mathcal{V}(\text{select\_flight}) = v$$

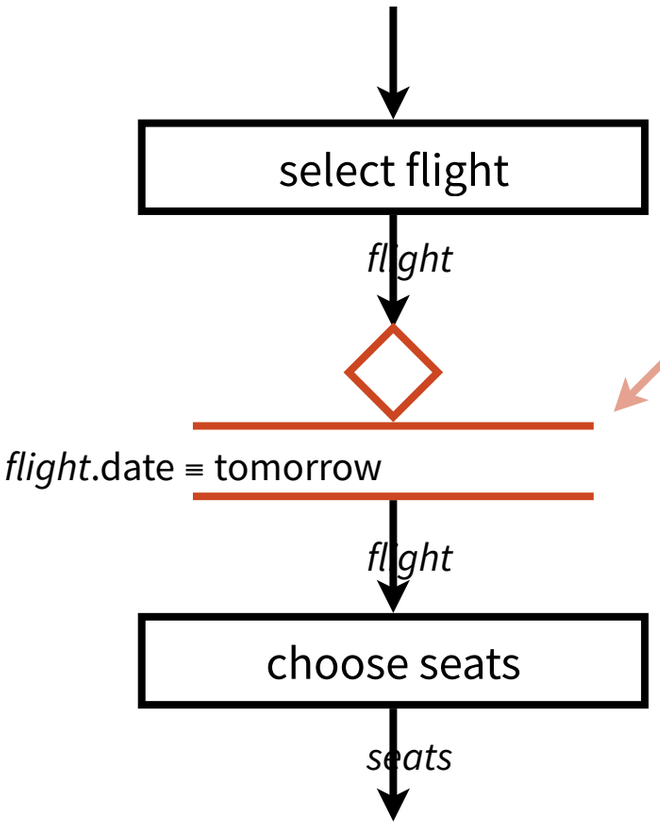
Guarded steps



ONLY WHEN GUARD IS TRUE?

select_flight ▸ λ*flight*. choose_seats *flight*

Guarded steps

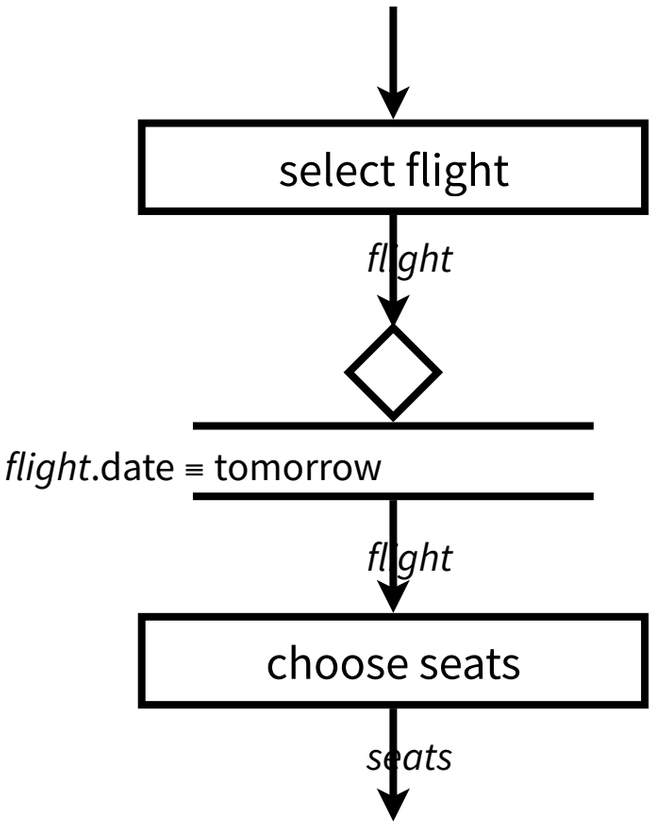


ONLY WHEN GUARD IS TRUE?

select_flight ▸ λ*flight*.
if flight.date ≡ tomorrow
then choose_seats *flight*
else ↯

FAILURE:
A TASK THAT NEVER ENDS
AND CAN NOT HANDLE USER INPUT

Guarded steps



select_flight ▸ $\lambda flight.$
if flight.date ≡ tomorrow
then choose_seats *flight*
else ↯

WHEN TO PROCEED TO THE NEXT TASK?

$\Longrightarrow \mathcal{V}(\text{select\_flight}) = v$
and … $(v) \Downarrow t$ **where** $t \neq \lightning$

USING HOST LANGUAGE SEMANTICS!

Institute for Computing and Information Sciences
Radboud University

Nitty gritty

Institute for Computing
and Information Sciences
Radboud University

Grammar

Take a λ-calculus…

$$
\begin{array}{lll}
e ::= & & \text{Expressions} \\
\quad | & \lambda x : \tau.\, e \mid e_1\ e_2 & \text{– abstraction, application} \\
\quad | & x \mid c \mid e_1 \star e_2 & \text{– variable, constant, operation} \\
\quad | & \textbf{if}\ e_1\ \textbf{then}\ e_2\ \textbf{else}\ e_3 \mid \langle\rangle & \text{– branch, unit} \\
\quad | & \langle e_1, e_2\rangle \mid \mathsf{fst}\, e \mid \mathsf{snd}\, e & \text{– pair, projections} \\
\quad | & \textbf{ref}\, e \mid\, !e \mid e_1 := e_2 \mid l & \text{– references, location} \\
\quad | & p & \text{– pretask} \\
c ::= & & \text{Constants} \\
\quad | & B \mid I \mid S & \text{– boolean, integer, string}
\end{array}
\qquad \longrightarrow \qquad
\begin{array}{lll}
t ::= & & \text{Tasks} \\
\quad | & \square\, v \mid \boxtimes\, \tau \mid \blacksquare\, l & \text{– editors} \\
\quad | & t_1 \blacktriangleright e_2 \mid t_1 \triangleright e_2 & \text{– steps} \\
\quad | & \lightning \mid t_1 \bowtie t_2 & \text{– fail, combination} \\
\quad | & t_1 \blacklozenge t_2 \mid e_1 \lozenge e_2 & \text{– choices}
\end{array}
$$

…embed a workflow language

GRAMMAR IS LAYERS, SO SEMANTICS ARE TO!

COMBINATION OF TWO TASKS

RACE BETWEEN TWO TASKS

Grammar

Take a λ-calculus…

$e ::=$		Expressions
	$\lambda x : \tau.\ e \mid e_1\ e_2$	– abstraction, application
	$x \mid c \mid e_1 \star e_2$	– variable, constant, operation
	$\textbf{if}\ e_1\ \textbf{then}\ e_2\ \textbf{else}\ e_3 \mid \langle\rangle$	– branch, unit
	$\langle e_1, e_2\rangle \mid \mathsf{fst}\ e \mid \mathsf{snd}\ e$	– pair, projections
	$\textbf{ref}\ e \mid !e \mid e_1 := e_2 \mid l$	– references, location
	p	– pretask
$c ::=$		Constants
	$B \mid I \mid S$	– boolean, integer, string

…embed a workflow language

$t ::=$		Tasks
	$\square\, v \mid \boxtimes\, \tau \mid \blacksquare\, l$	– editors
	$t_1 \blacktriangleright e_2 \mid t_1 \triangleright e_2$	– steps
	$\lightning \mid t_1 \bowtie t_2$	– fail, combination
	$t_1 \blacklozenge t_2 \mid e_1 \lozenge e_2$	– choices

GRAMMAR IS LAYERS, SO SEMANTICS ARE TO!

COMBINATION OF TWO TASKS

RACE BETWEEN TWO TASKS

Institute for Computing and Information Sciences
Radboud University

Semantics

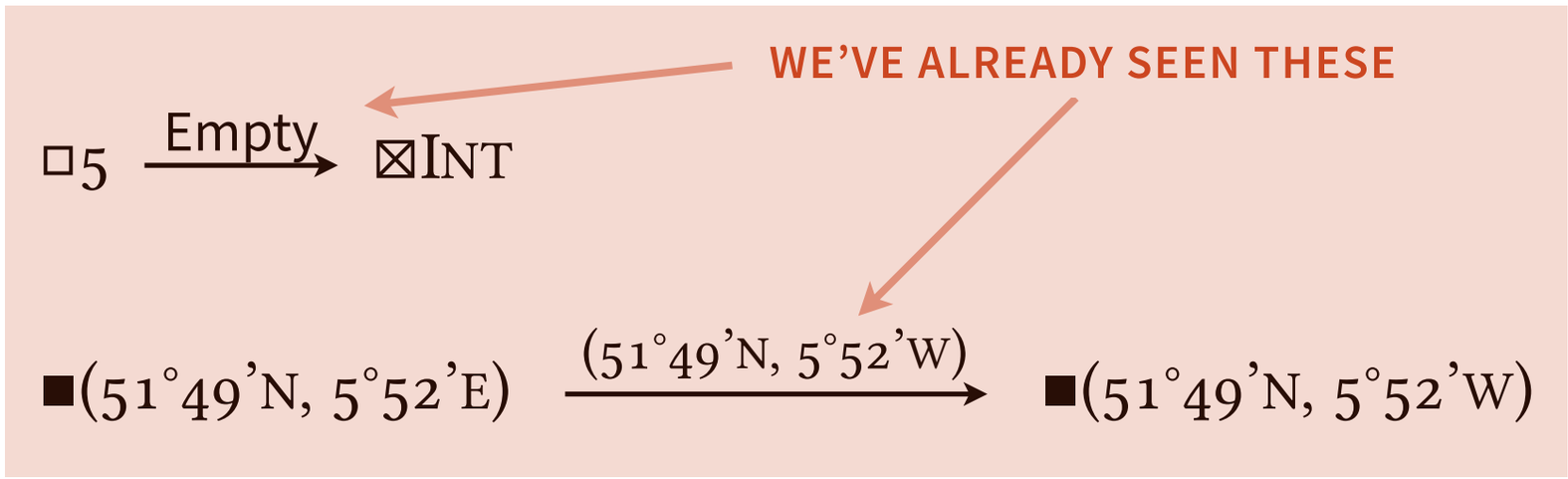
Two layers ⇒ two semantics

$e \downarrow v$ STANDARD BIG STEP SEMANTICS

$p \Downarrow t$ SPECIAL TASK SEMANTICS

But interaction… ⇒ additional layer

$t \xrightarrow{i} t'$



Comparisons

Institute for Computing
and Information Sciences
Radboud University

Tasks vs Processes

TOP	CSP
“We do *A* together, then you do *B*”	“I send you *M*, you receive it and react wit *N*”
Developer models collaboration	Developer models communication
System takes care of communication	

Future work

Summary

Extras

Institute for Computing
and Information Sciences
Radboud University