School of informatics

A theorem proving framework for the formal verification of Web Services Composition

Petros Papapanagiotou

WWV 2011

7th Int'l Workshop on Automated Specification and Verification of Web Systems, Reykjavik, Iceland
June 9, 2011

Web Services Composition

- Mechanically combine Web Services for complex tasks

- In this work:
  - Offline
  - Non-functional properties (Quality driven)
  - Exceptions
  - Formal

WS Composition Methodologies

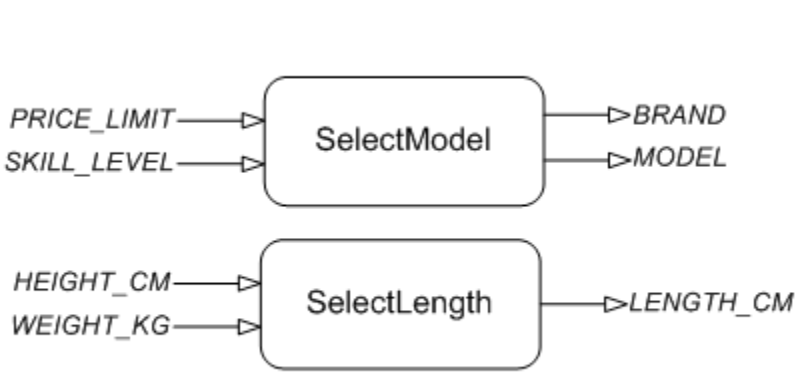
- Workflow techniques (eg. EFlow)
- AI Planning (eg. SWORD)
- Theorem Proving

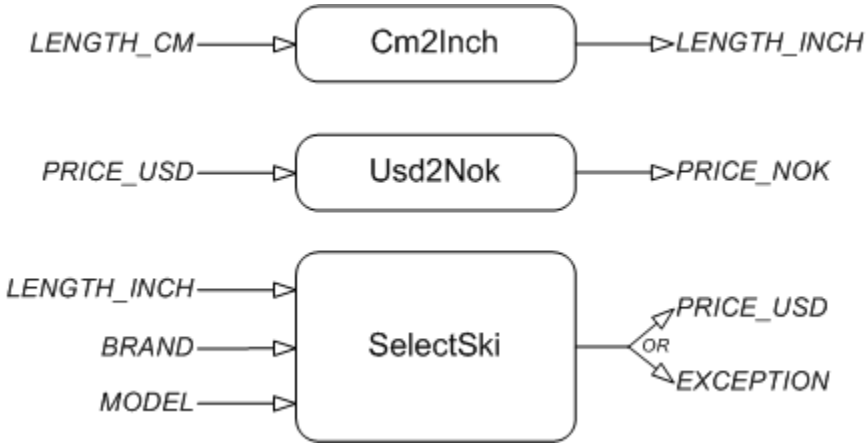
WS Composition using Theorem Proving

- Waldinger (2001):
  - program synthesis in SNARK - FOL
- Lammermann (2002):
  - structural synthesis in Java – Intuitionistic Propositional Logic
- Rao et al. (2006):
  - proofs-as-processes in RAPS – ILL
- Current work:
  - proofs-as-processes in HOL Light – CLL

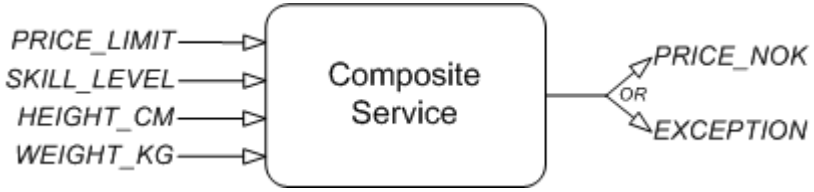
Ski Example

- Ski purchasing (Rao 2004)
- Available services:





- Requested service:



Proofs-as-processes

- Similar to "formulae-as-types" (Curry-Howard 1980)
- Introduced by Abramsky (1994)
- Bellin and Scott (1994)
- Classical Linear Logic and π-calculus

Classical Linear Logic

- Girard (1987)

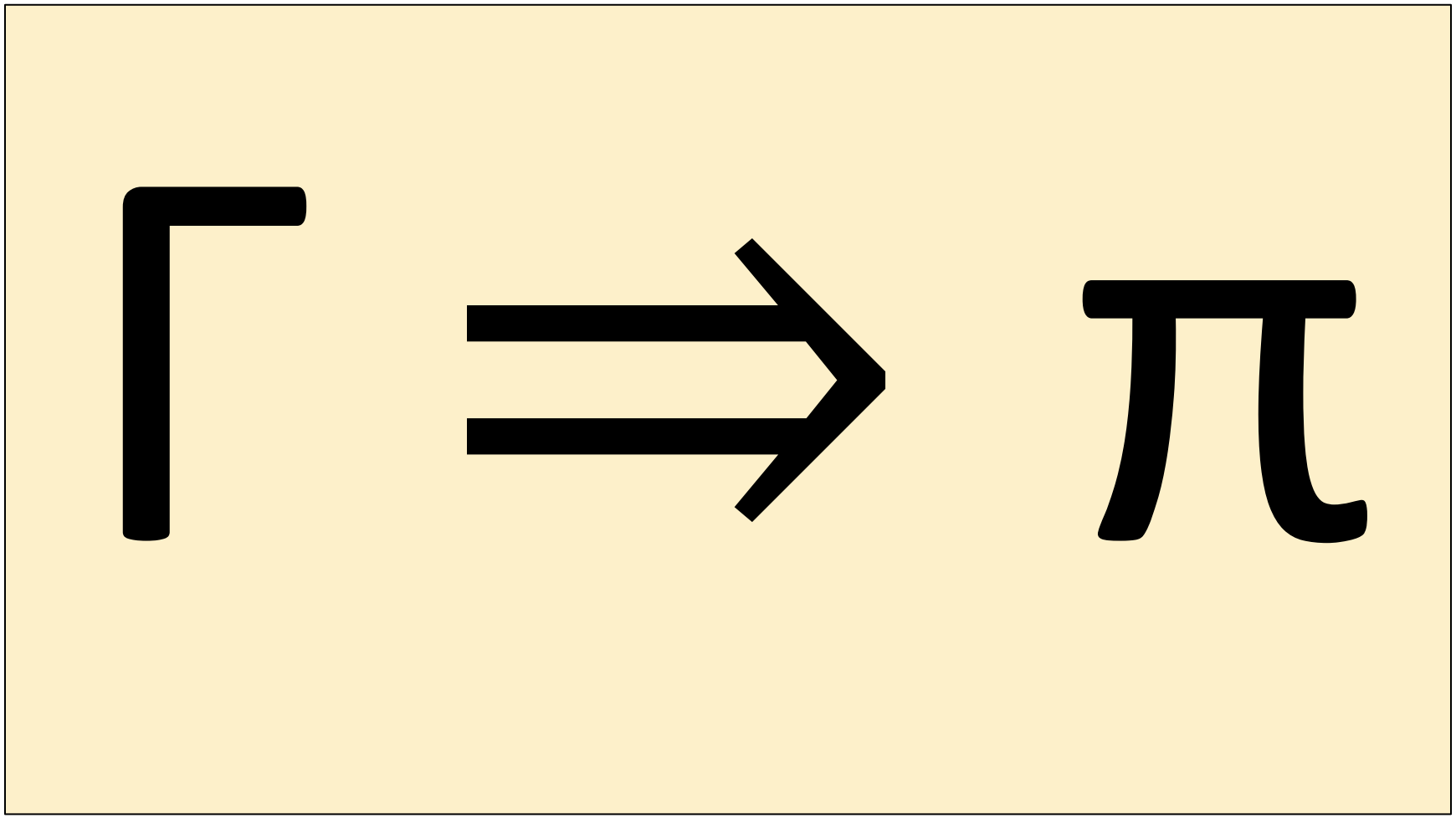
	Disjunction	Conjunction
Multiplicative	$\parr$	$\otimes$
Additive	$\oplus$	$\&$
Negation	$(\cdot)^{\perp}$	

π-calculus

- Milner (1989)

P ::=	
| 0	null process
| x(y).P	input
| $\bar{x}$<y>.P	output
| (ν x) P	local variable
| P || P	parallel processes
| P + P	choice

Proofs-as-processes



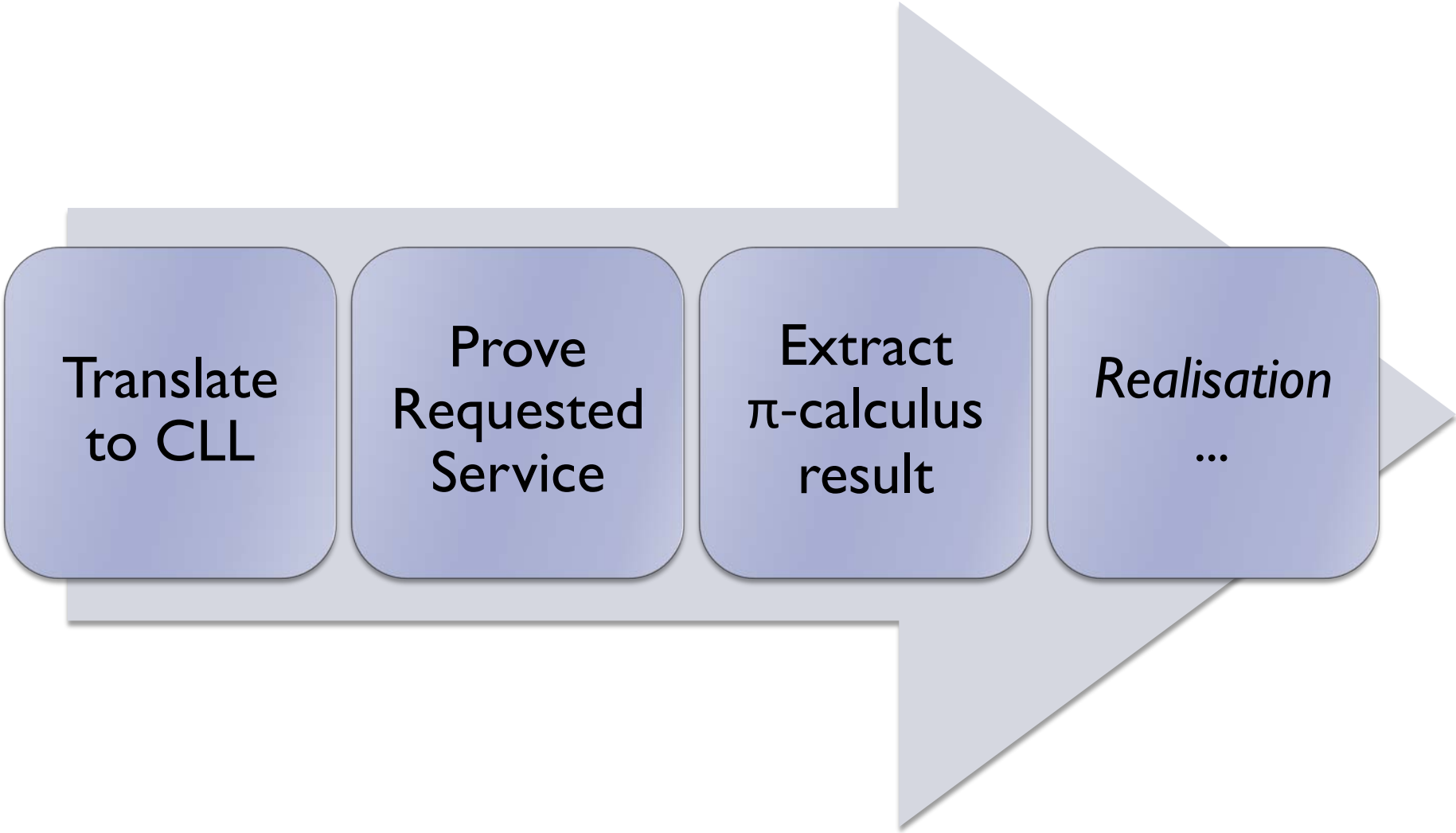
Example: Tensor ($\otimes$) rule

$$\frac{\begin{array}{cc} \vdots\ F & \vdots\ G \\ \vdash \vec{w}:\Gamma,\ x:A & \vdash \vec{u}:\Delta,\ y:B \end{array}}{\vdash \vec{w}:\Gamma,\ \vec{u}:\Delta,\ z:A \otimes B}\ \otimes$$

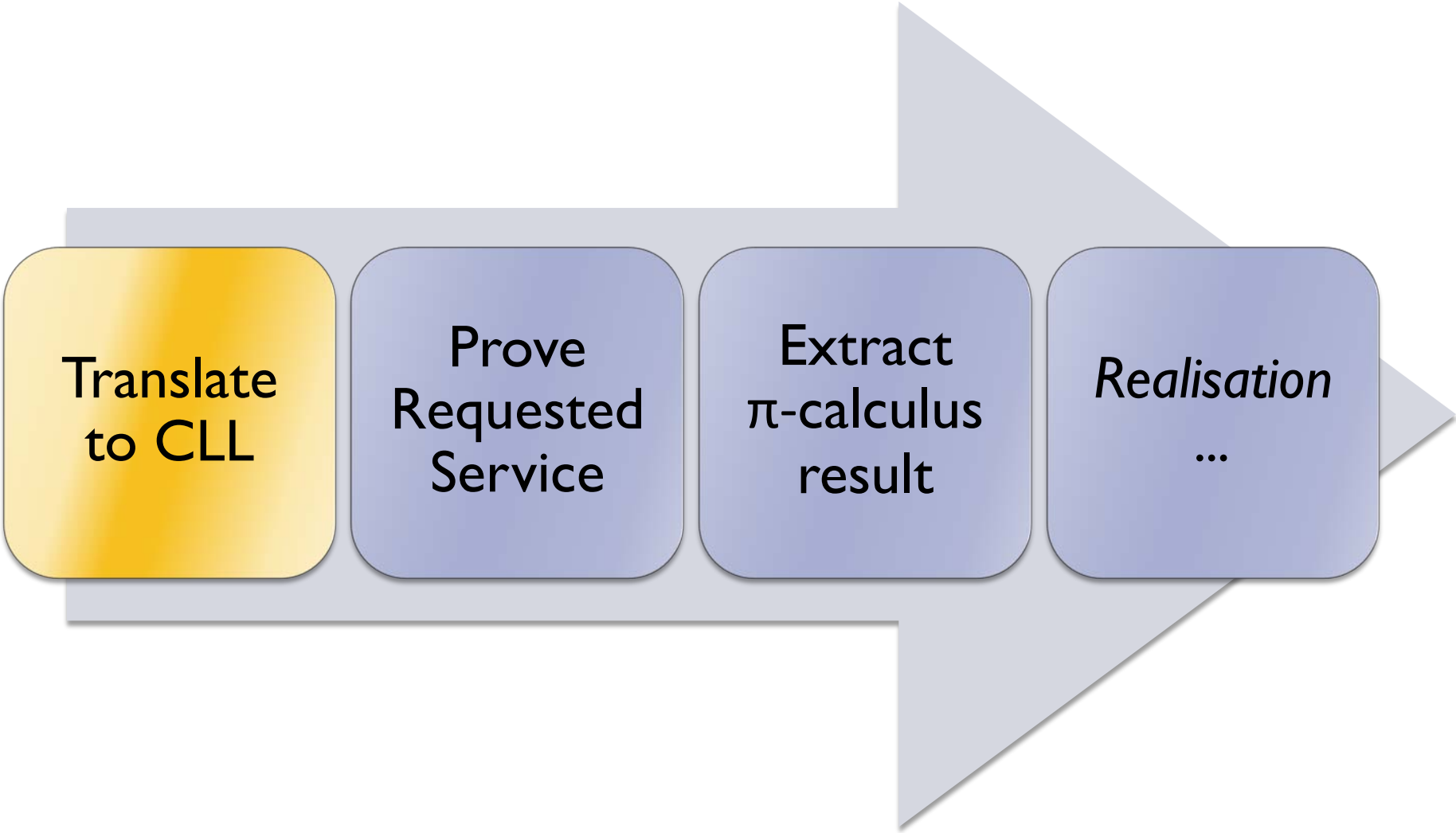
$$\bigotimes_{z}^{x,y}(F,G)\vec{w}\vec{u}z \equiv \nu xy(\overline{z}\langle xy\rangle(F_{\vec{w}x}||G_{\vec{u}y}))$$

CLL inference rule	π-calculus translation
$\vdash x:A,\ y:A^{\perp}$	$Ixy = y(a)\overline{x}\langle a\rangle$
$\dfrac{\vdots\ F \qquad \vdots\ G}{\vdash \vec{w}:\Gamma,\ x:A \quad \vdash \vec{u}:\Delta,\ y:B}\ \otimes$	$\overset{x,y}{\underset{z}{\bigotimes}}(F,G)\vec{w}\vec{u}z = \nu xy(\overline{z}\langle xy\rangle(F_{\vec{w}x}\|\|G_{\vec{u}y}))$
$\dfrac{\vdots\ F}{\vdash \vec{w}:\Gamma,\ x:A,\ y:B}\ \parr$	$\overset{x,y}{\underset{z}{\parr}}(F)\vec{w}z = z(xy)F_{\vec{w}xy}$
$\dfrac{\vdots\ P}{\vdash \vec{w}:\Gamma,\ x:A}\ \oplus$	$\overset{x}{\underset{z}{\mathrm{L}}}(P)\vec{w}z = \nu x(z(uv)\overline{u}\langle x\rangle P_{\vec{w}x})$
$\dfrac{\vdots\ Q}{\vdash \vec{w}:\Gamma,\ y:B}\ \oplus$	$\overset{y}{\underset{z}{\mathrm{R}}}(Q)\vec{w}y = \nu y(z(uv)\overline{v}\langle y\rangle Q_{\vec{w}y})$
$\dfrac{\vdots\ P \qquad \vdots\ Q}{\vdash \vec{w}:\Gamma,\ x:A \quad \vdash \vec{w}:\Gamma,\ y:B}\ \&$	$\overset{x,y}{\underset{z}{\&}}(P,Q)\vec{w}z = \nu uv(\overline{z}\langle uv\rangle[u(x)P_{\vec{w}x} + v(y)Q_{\vec{w}y}])$
$\dfrac{\vdots\ F \qquad \vdots\ G}{\vdash \vec{u}:\Gamma,\ x:C \quad \vdash \vec{v}:\Delta,\ y:C^{\perp}}\ Cut$	$Cut^{z}(F,G)\vec{u}\vec{v} = \nu z(F_{\vec{u}}[z/x]\|\|G_{\vec{v}}[z/y])$

WS Composition using proofs-as-processes



WS Composition using proofs-as-processes



Web Services in Classical Linear Logic

$$\vdash \vec{P}^{\perp}, \vec{I}^{\perp}, ((\otimes_{i} \vec{F}) \otimes (\otimes_{i} \vec{O})) \oplus E$$

$$\text{Where}: \otimes_{i}(\alpha_1, \alpha_2, ..., \alpha_n) = \alpha_1 \otimes \alpha_2 \otimes ... \otimes \alpha_n \ , \ i \geq 0$$

$\vec{P}$: Preconditions

$\vec{I}$: Input

$\vec{F}$: Postconditions (effects)

$\vec{O}$: Output

$\vec{E}$: Exception

Ski example specified in CLL

- `SelectModel`:

  $\vdash \text{PRICE\_LIMIT}^{\perp}, \text{SKILL\_LEVEL}^{\perp}, \text{BRAND} \otimes \text{MODEL}$

- `SelectLength`:

  $\vdash \text{HEIGHT\_CM}^{\perp}, \text{WEIGHT\_KG}^{\perp}, \text{LENGTH\_CM}$

- `Cm2Inch`:

  $\vdash \text{LENGTH\_CM}^{\perp}, \text{LENGTH\_IN}$

- `Usd2Nok`:

  $\vdash \text{PRICE\_USD}^{\perp}, \text{PRICE\_NOK}$

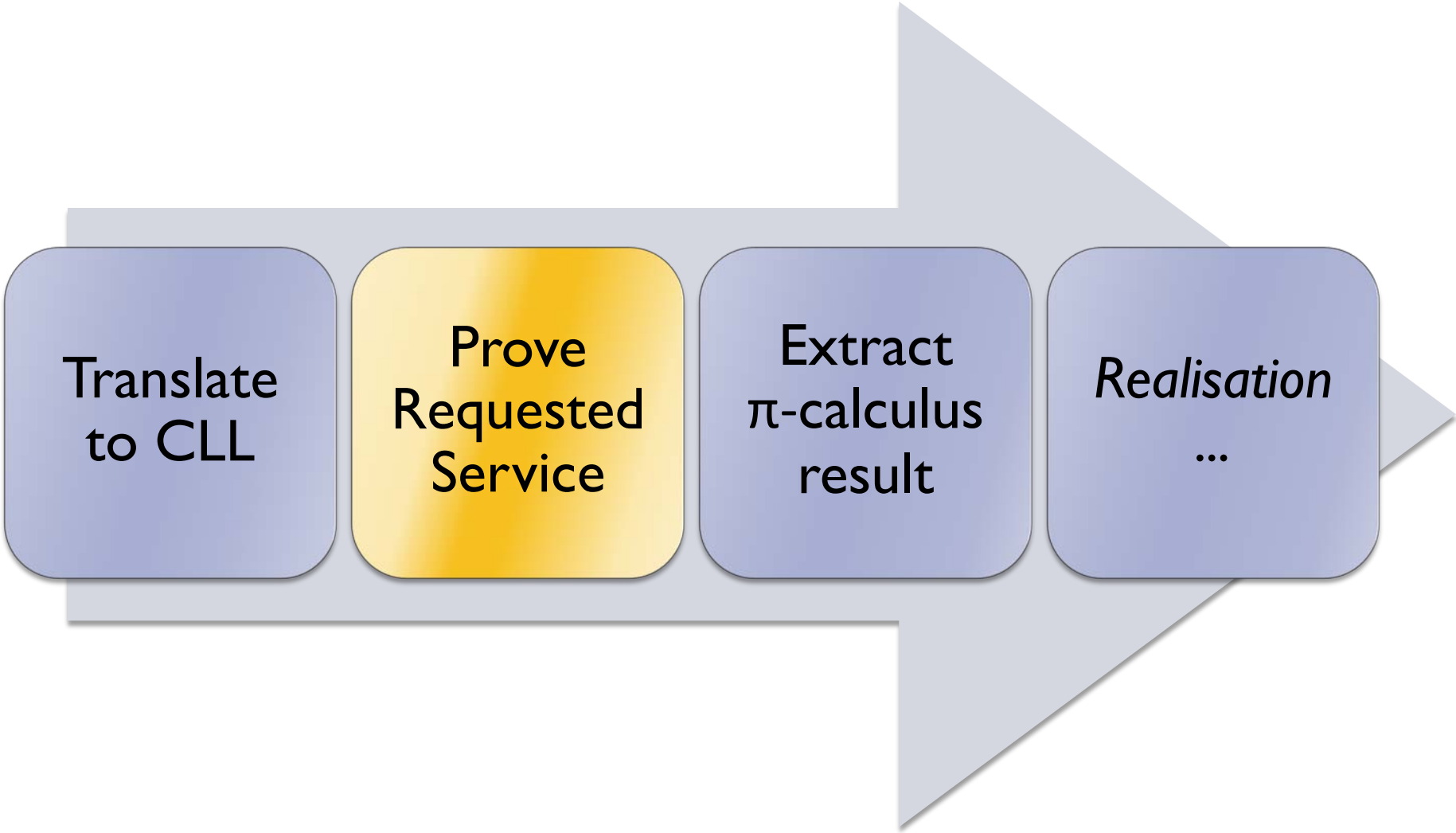
- `SelectSki`:

  $\vdash \text{LENGTH\_IN}^{\perp}, \text{BRAND}^{\perp}, \text{MODEL}^{\perp}, \text{PRICE\_USD} \oplus \text{EXCEPTION}$

Ski Request in CLL

$$\vdash \text{PRICE\_LIMIT}^{\perp},\ \text{SKILL\_LEVEL}^{\perp},\ \text{HEIGHT\_CM}^{\perp},\ \text{WEIGHT\_KG}^{\perp},\ \text{PRICE\_NOK} \oplus \text{EXCEPTION}$$

WS Composition using proofs-as-processes



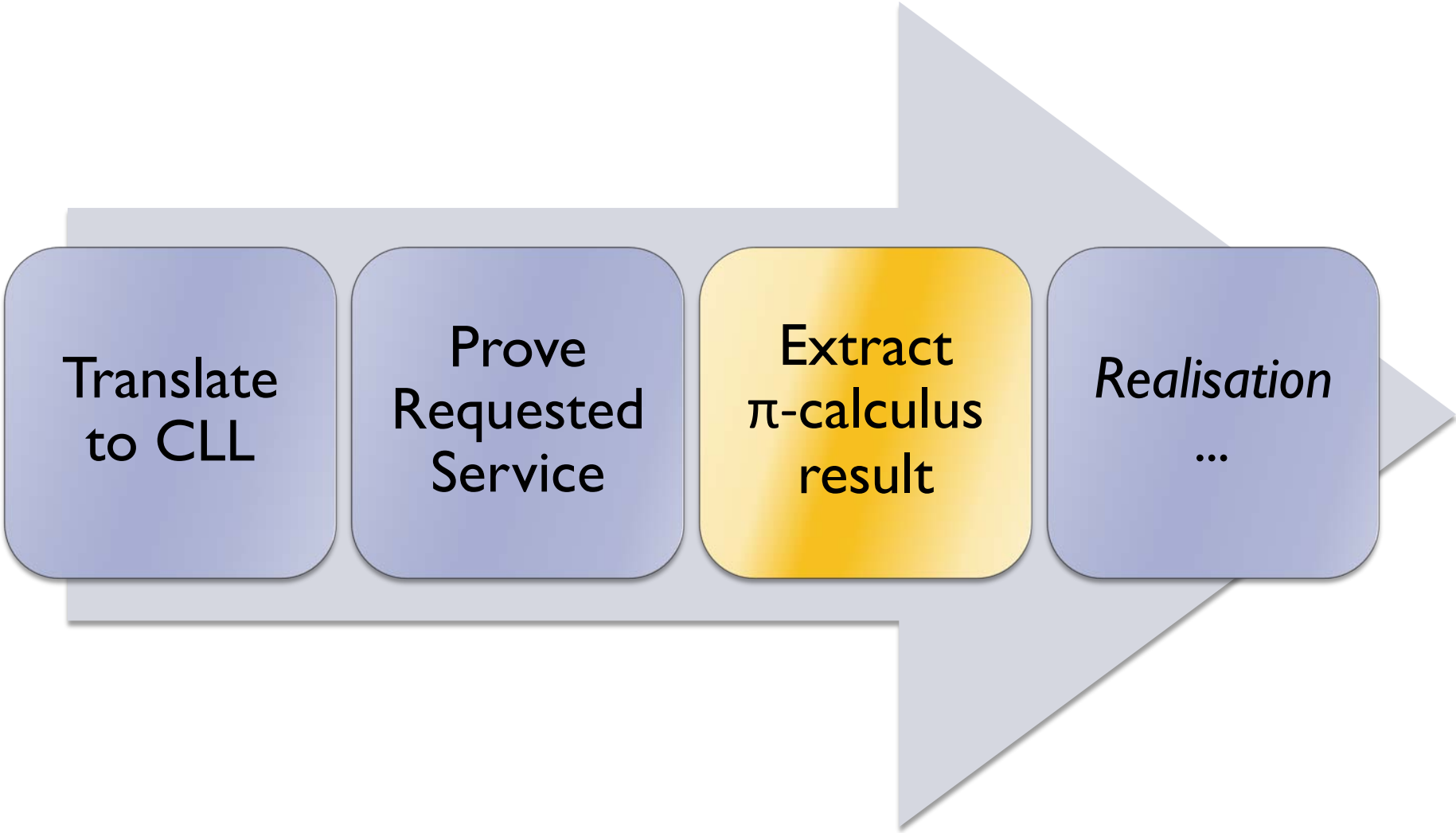
Proof for the Ski example

$$\dfrac{\dfrac{}{\vdash PL^{\perp},\ SL^{\perp},\ BR \otimes MO}\ SelMod \qquad \dfrac{\dfrac{}{\vdash HC^{\perp},\ WK^{\perp},\ LC}\ SelLen \qquad \dfrac{}{\vdash LC^{\perp},\ LI}\ Cm2Inch}{\vdash HC^{\perp},\ WK^{\perp},\ LI}\ Cut}{\vdash PL^{\perp},\ SL^{\perp},\ HC^{\perp},\ WK^{\perp},\ (BR \otimes MO) \otimes LI}\ \otimes \qquad (1)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash BR^{\perp},\ MO^{\perp},\ LI^{\perp},\ PU \oplus EXE}\ SelSki}{\vdash BR^{\perp} \parr MO^{\perp},\ LI^{\perp},\ PU \oplus EXE}\ \parr}{\vdash (BR^{\perp} \parr MO^{\perp}) \parr LI^{\perp},\ PU \oplus EXE}\ \parr \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash PU^{\perp},\ PN}\ Usd2Nok}{\vdash PU^{\perp},\ PN \oplus EXE}\ \oplus \qquad \dfrac{\dfrac{}{\vdash EXE^{\perp},\ EXE}\ Id}{\vdash EXE^{\perp},\ PN \oplus EXE}\ \oplus}{\vdash PU^{\perp} \,\&\, EXE^{\perp},\ PN \oplus EXE}\ \&}{\vdash (PU \oplus EXE)^{\perp},\ PN \oplus EXE}\ neg\_eq}{\vdash (BR^{\perp} \parr MO^{\perp}) \parr LI^{\perp},\ PN \oplus EXE}\ Cut \qquad \text{(2)}$$

$$\dfrac{\dfrac{\vdash (BR^{\perp} \parr MO^{\perp}) \parr LI^{\perp},\ PN \oplus EXE}{\vdash ((BR \otimes MO) \otimes LI)^{\perp},\ PN \oplus EXE}\ neg\_eq}{\vdash PL^{\perp},\ SL^{\perp},\ HC^{\perp},\ WK^{\perp},\ PN \oplus EXE}\ Cut\ with\ (1)$$

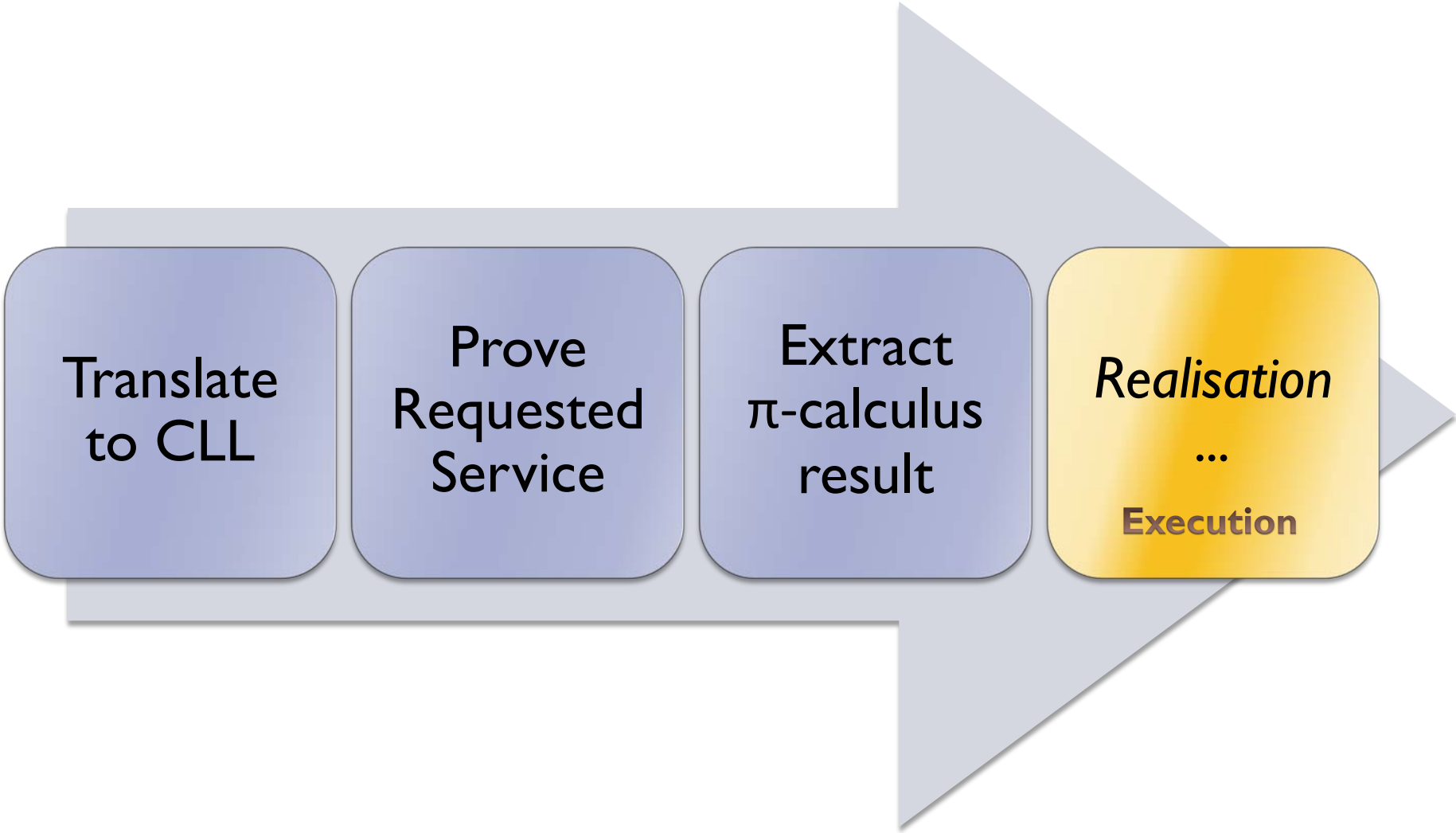
WS Composition using proofs-as-processes



Ski Result in π-calculus

$$
\begin{aligned}
&Composition(smp, sms, slh, slw, t, puc, exc) \equiv \\
&\quad (\nu\ z_1) \\
&\qquad ((\nu\ smo, cii) \\
&\qquad\quad (\overline{z_1}\langle smo, cii\rangle. \\
&\qquad\quad (SelMod \mid\mid (\nu\ z_3)(SelLen[z_3/sll] \mid\mid Cm2Inch[z_3/cic])) \\
&\qquad) \mid\mid (\nu\ z_4) \\
&\qquad\quad (((z_1(x_5, ssl).x_5(ssb, ssm).SelSki[z_4/sso]) \mid\mid \\
&\qquad\quad (\nu\ u_7, v_7) \\
&\qquad\qquad ((\overline{z_4}\langle (u_7, v_7)\rangle. \\
&\qquad\qquad\quad ((u_7(unu).(\nu\ unn)(t(u_8, v_8).\overline{u_8}\langle unn\rangle.Usd2Nok)) + \\
&\qquad\qquad\qquad (v_7(y_7).(\nu\ y_9)(t(u_9, v_9).\overline{v_9}\langle y_9\rangle.y_7(a_{10}).\overline{y_9}\langle a_{10}\rangle.0))) \\
&\qquad\quad)) \\
&\qquad) \\
&\quad)
\end{aligned}
$$

WS Composition using proofs-as-processes



Execution

- Available services are "black-boxes"

- Empirical translations:

$$
\begin{array}{rl}
A & \overline{a}\langle\ldots\rangle.0 \\
A^{\perp} & a(\ldots).0 \\
A \otimes B & (\nu a,b)(\overline{z}\langle a,b\rangle.(\overline{a}\langle\ldots\rangle.0 \mid\mid \overline{b}\langle\ldots\rangle.0)) \\
A^{\perp} \parr B^{\perp} & z(a,b).(a(\ldots).0 \mid\mid b(\ldots).0) \\
A \oplus B & (\nu a,b)(z(u,v).(\overline{u}\langle x\rangle.\overline{x}\langle\ldots\rangle.0 + \overline{v}\langle y\rangle.\overline{y}\langle\ldots\rangle.0)) \\
A^{\perp} \,\&\, B^{\perp} & (\nu a,b)(\overline{z}\langle u,v\rangle.(u(x).x(\ldots).0 + v(y).y(\ldots).0))
\end{array}
$$

Ski example specified in π-calculus

$$SelLen(slh, slw, sll, lc) \equiv slh(hc).slw(wk).\overline{sll}\langle lc\rangle.0$$

$$Cm2In(cic, cli, li) \equiv cic(lc).\overline{cli}\langle li\rangle.0$$

$$Usd2Nok(unu, unn, pn) \equiv unu(pu).\overline{unn}\langle pn\rangle.0$$

$$SelMod(smp, sms, smm, br, mo) \equiv smp(pl).sms(sl).(\nu\ smb, smo)(\overline{smm}\langle smb, smo\rangle.\overline{smb}\langle br\rangle.\overline{smo}\langle mo\rangle.0)$$

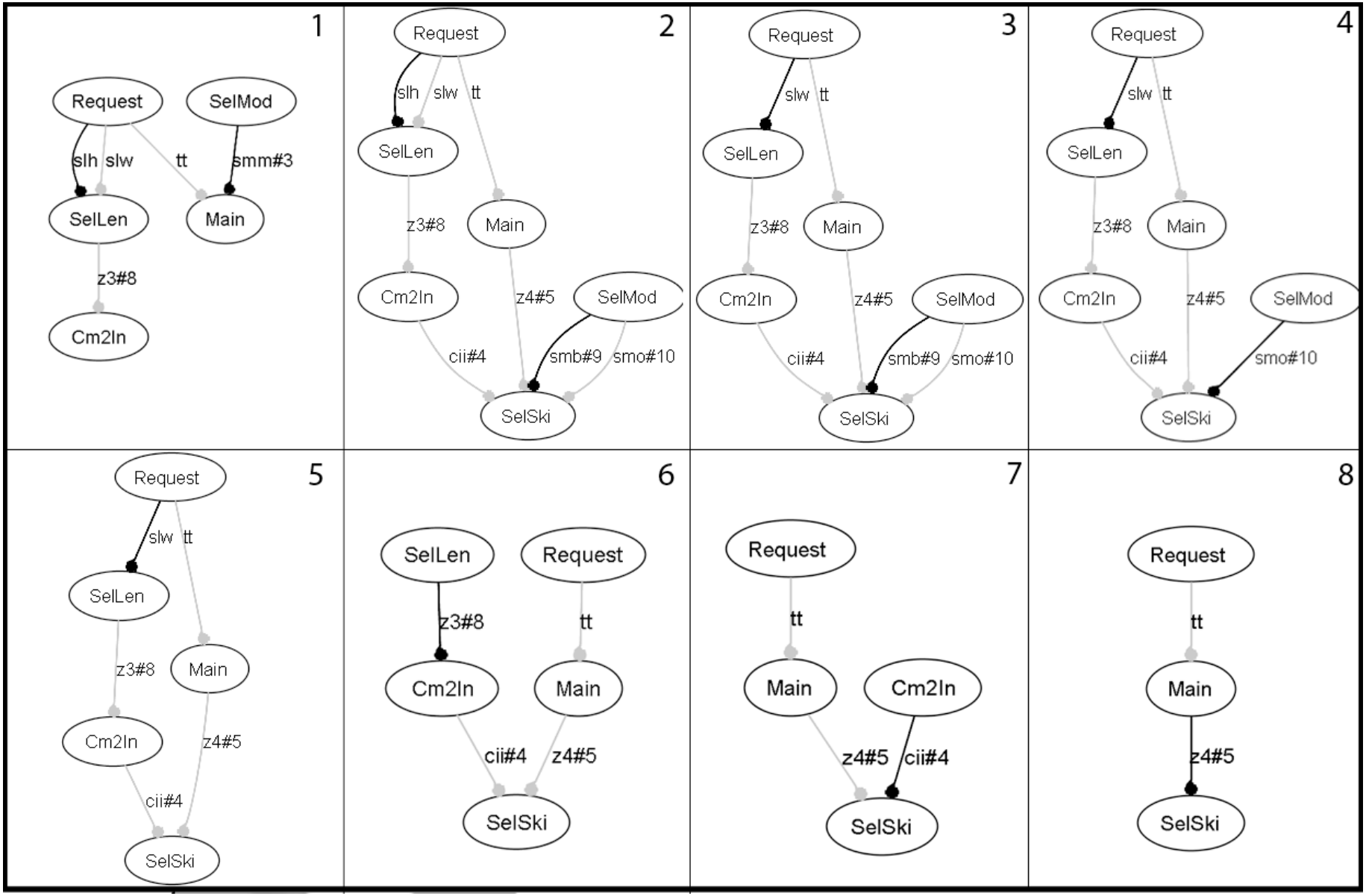
$$SelSki(ssb, ssm, ssl, sso, pu, ex) \equiv (\nu\ ssp, sse)(ssb(br).ssm(mo).ssl(li).sso(u, v).(\overline{u}\langle ssp\rangle.\overline{ssp}\langle pu\rangle.0 + \overline{v}\langle sse\rangle.\overline{sse}\langle ex\rangle.0))$$

$$Request(smp, pl, sms, sl, slh, hc, slw, wk, t, puc, exc) \equiv \overline{smp}\langle pl\rangle.\overline{sms}\langle sl\rangle.\overline{slh}\langle hc\rangle.\overline{slw}\langle wk\rangle.\overline{t}\langle puc, exc\rangle.(puc(x).x(pu).0 + exc(y).y(ex).0)$$

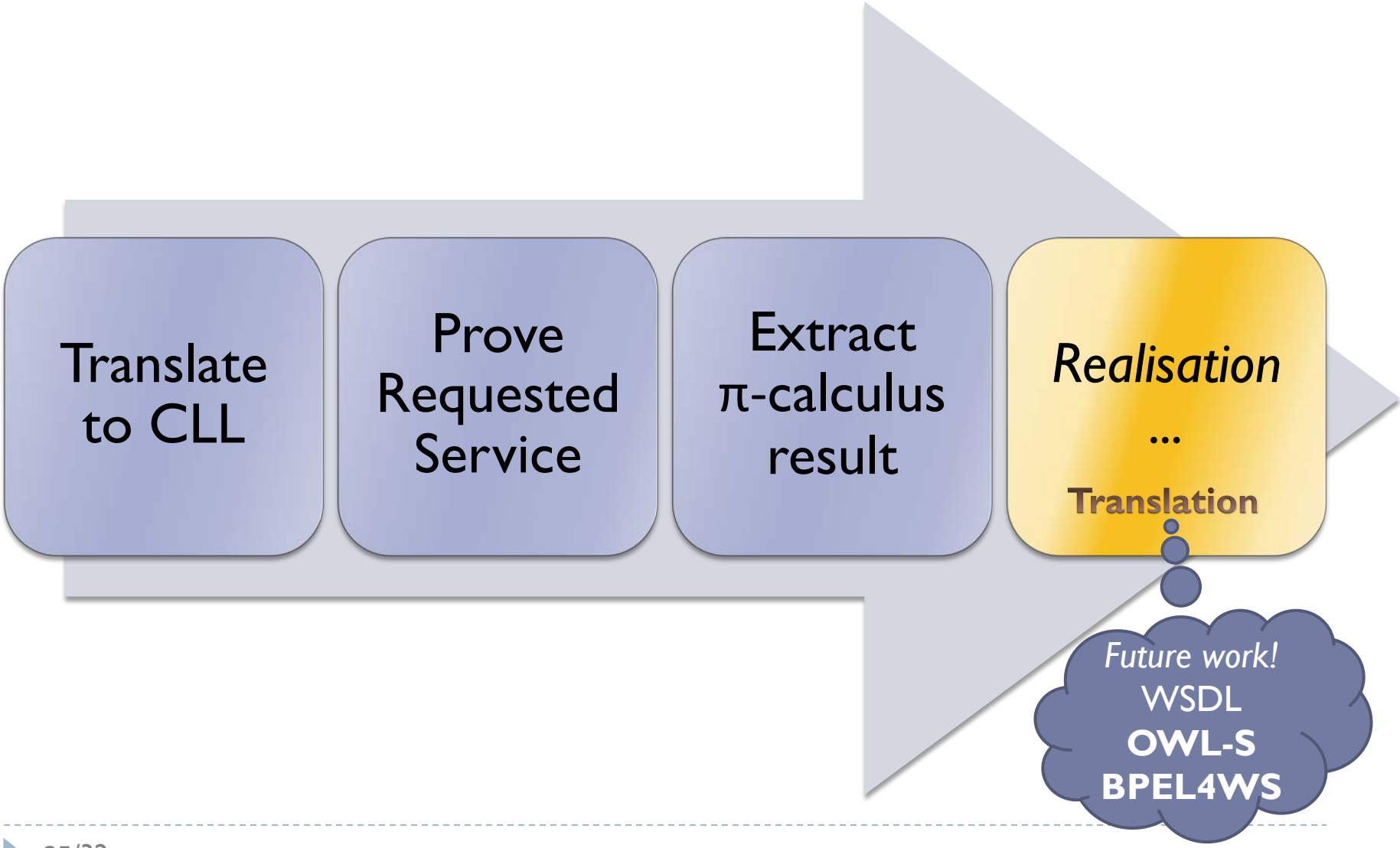
$$Main() \equiv Request(smp, pl, sms, sl, slh, hc, slw, wk, t, puc, exc) \mid\mid Composition(smp, sms, slh, slw, t, puc, exc)$$

PiVizTool

- Bog (2006)
- Visualisation of connections
- Animation of execution
- Empirical verification


1
Request
SelMod
slh
slw
tt
smm#3
SelLen
Main
z3#8
Cm2In
2
Request
slh
slw
tt
SelLen
Main
z3#8
Cm2In
z4#5
SelMod
cii#4
smb#9
smo#10
SelSki
3
Request
slw
tt
SelLen
Main
z3#8
Cm2In
z4#5
SelMod
cii#4
smb#9
smo#10
SelSki
4
Request
slw
tt
SelLen
Main
z3#8
Cm2In
z4#5
SelMod
cii#4
smo#10
SelSki
5
Request
slw
tt
SelLen
Main
z3#8
Cm2In
z4#5
cii#4
SelSki
6
SelLen
Request
z3#8
tt
Cm2In
Main
cii#4
z4#5
SelSki
7
Request
tt
Main
Cm2In
z4#5
cii#4
SelSki
8
Request
tt
Main
z4#5
SelSki

WS Composition using proofs-as-processes



Implementation: Requirements

- Deep embedding
  - CLL rules + proofs-as-processes translations
- Theory + automation of reasoning for multisets
  - No need for Exchange rule:

$$\frac{\vdash \mathrm{A}, B, \Gamma, \Delta}{\vdash \mathrm{A}, \Gamma, B, \Delta}\, Exchange$$

- Metavariables
  - Construction of π-calculus translations
  - Synthesis of exceptions

Implementation: Details

- HOL Light – flexible, programmable
- Isabelle Light – procedural proofs, metavariables

CLL

- Inspired by Power et al. (1999) and Sadrzadeh (2003)
- Conservative
- Combined inference rules – proofs-as-processes

π-calculus

- Based on Melham (1992)
- Syntax (polymorphic type)
- Substitution
- A few functions

Implementation: π-calculus

P ::=	**define_type (A) Agent =**
| 0	Zero
| x(y).P	| In A (A list) Agent
| x<y>.P	| Out A (A list) Agent
| (ν x) P	| Res (A list) Agent
| P || P	| Comp Agent Agent
| P + P	| Plus Agent Agent

Implementation: CLL

CLL connective	HOL Light definition	HOL Light syntax
!	`OfCourse`	`!!`
?	`WhyNot`	`??`
$\otimes$	`LinTimes`	`**`
$\parr$	`LinPar`	`%`
$\oplus$	`LinPlus`	`++`
&	`LinWith`	`&`

Implementation: Proofs-as-processes

$$\frac{\vdash P :: \vec{w}:\Gamma,\ x:A \qquad \vdash Q :: \vec{u}:\Delta,\ y:B}{\vdash \nu xy(\overline{z}\langle xy\rangle(P_{\vec{w}x} \,||\, Q_{\vec{u}y})) :: \vec{w}:\Gamma,\ \vec{u}:\Delta,\ z:A \otimes B}\ \otimes$$

```
!x y z P Q A B Gamma Delta.
 |-- (Gamma ^ ' (A <> x)) P /\ |-- (Delta ^ ' (B <> y)) Q ==>
  |-- (Gamma ^ Delta ^ ' ((A ** B) <> z)) (Res [x;y] (Out z [x;y] (Comp P Q)))
```

Future Work

- Translation:
  - From OWL-S/BPEL4WS to CLL.
  - From π-calculus to OWL-S/BPEL4WS.
- Automation
  - External + internal tools
- Further Evaluation
  - Already a case-study involving 10 WS - real-estate domain
  - More examples
- Embedding
  - Modalities

Conclusion

- Formally verified Web Services composition.
  - Based on the proofs-as-processes paradigm.
  - Using CLL + π-calculus.
  - Implemented in HOL Light
- Rao's Ski example

- Contribution:
  - Web Services community: Rigorous WS composition
  - Theorem Proving community: Framework for theoretical investigation
- Much room for further development.