

Type Directed Interactive Workflow Modelling

Tim Steenvoorden

Rinus Plasmeijer

Dutch FP Day, January 5, 2018



Example

```
requestSolarPanelCompensation :: Citizen -> Task ()
requestSolarPanelCompensation citizen
=
  checkConditions citizen
>>- \checks -> if (not checks.ownsRealEstate || not checks.noSubsidyPast5Years)
  (showChecks checks)
(
  obtainDeclarations citizen
>>- \result ->
case result of
  CanceledByCitizen _ = return ()
  CanceledByCompany _ = showChecks {checks & declarationCompany = False}
  Declarations dossier = submitOrCancelSubsidy dossier)
```

Request solar panel subsidy

1. present a webform to an applicant
2. check if he/she is obligable for a subsidy
3. ask for tax compensation documents and a contractor declaration
4. submit the request to the tax office

```
:: Declarations = CanceledByCitizen NameHomeAddress
                 | CanceledByCompany Company
                 | Declarations TaxSolarPanelDossier

obtainDeclarations :: Citizen -> Task Declarations
obtainDeclarations citizen
=
  get currentDate
>>- \d -> deadlineWith (clientDeadlineDate d) Nothing
  (maybeCancel "Cancel_Request"
   (declarationApplicant d -&&- declarationCompany d applicant))
@ toDeclarations d applicant
where applicant = nameHomeAddressFromCitizen citizen
```

```
provideDeclaration :: Date NameHomeAddress Company -> Task (Maybe CompanyDeclaration)
provideDeclaration today applicant company
= viewInformation msg [] applicant
  >>* [ OnAction (Action "Yes,I provide declaration")
        (always (provide >>- return o Just))
      , OnAction (Action "No,unknown customer") (always (return Nothing))]
where
  msg = "This customer would like to receive a declaration for the tax authorities:"
```

```
submitOrCancelSubsidy :: TaxSolarPanelDossier -> Task ()
submitOrCancelSubsidy dossier
= viewInformation "You can submit the subsidy" [] dossier
  >>* [ OnAction (Action "Submit") (always (submitSubsidy dossier))
      , OnAction (Action "Cancel") (always (return ())) ]

submitSubsidy :: TaxSolarPanelDossier -> Task ()
submitSubsidy dossier
= get currentDate
  >>- \date -> let dossier = {dossier & date = date}
  in (( viewInformation "Your request is being processed" [] ())
      ||-
      ( (UserWithRole "officer","Subsidy_request") @: processRequest dossier)
  >>- \decision -> viewInformation "Your request has been processed" [] decision
  >>* [ OnAction (Action "Edit_request") (ifCond (decision.status <> Approved)
        (resubmitSubsidy dossier))
      , OnAction (Action "Cancel_request") (ifCond (decision.status <> Approved)
        (return ()))
      , OnAction (Action "Continue") (ifCond (decision.status == Approved)
        (return ())) ]
  >>| return ()

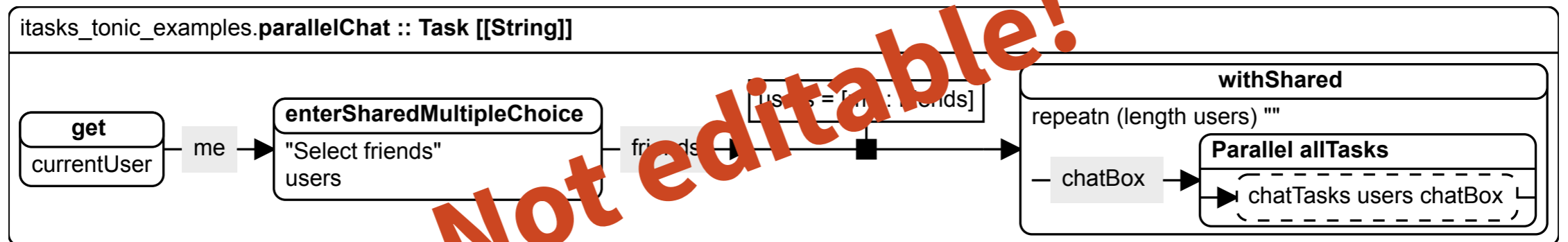
resubmitSubsidy :: TaxSolarPanelDossier -> Task ()
resubmitSubsidy dossier
= updateInformation "Edit your documents" [] dossier.request.documents
  >>= \new -> submitOrCancelSubsidy {dossier & request.documents = new }
```

```
declarationApplicant :: Date -> Task TaxCompensationDocuments
declarationApplicant today
= (enterInformation msg [] >>= return)
  -||
  (reminder (clientReminderDate today) "finish your request for tax compensation")
where
  msg = "Please enter the following information for your tax compensation request:"
```

```
declarationCompany :: Date NameHomeAddress -> Task (Company, Maybe CompanyDeclaration)
declarationCompany today applicant
= selectOfficialSolarPanelCompany
  >>- \company ->
  (company.cocNo,"Request declaration")
  @: (provideDeclaration today applicant company)
  -||
  (reminder (clientReminderDate today) "please finish the proof")
  >>= \decl -> viewInformation (msg_decision company decl) [] decl
  >>| return (company,decl)
where
  msg_decision c d = "Declaration was_" +++ if (isNothing d) "negative" "positive"
```

Goals

- Understandable by domain experts
- Comprehensible set of basic elements
- Visual representation



Tonic by Stutterheim (2014, 2015)

- Editable using handful of operations
- Assist during design of workflows

Today

What?

- One language (domain specific, subset of *iTasks*)
- Two representations:
 - Visual
 - Textual

How?

1. Take a problem
2. Solve it in easy steps
3. Correct our solution
4. Study some background



Visual Representation



Example

Request solar panel subsidy

1. **present a webform** to an applicant
2. **check** if he/she is obligable for a subsidy
3. ask for **tax compensation documents** and a **contractor declaration**
4. **submit** the request to the tax office



Library

provide declaration

uses *applicant* : CitizenInfo,
contractor : ContractorInfo
yields *declaration* : ContractorDecl

provide documents

uses –
yields *documents* : CompensationDocs

select contractor

uses –
yields *contractor* : ContractorInfo

submit request

uses *documents* : CompensationDocs,
declaration : ContractorDecl
yields –

request subsidy

Operations

Init

Extend ↑

Swap

Extend ↓

Split

Lift

Library

provide declaration

uses *applicant* : CitizenInfo,
contractor : ContractorInfo
yields *declaration* : ContractorDecl

provide documents

uses –
yields *documents* : CompensationDocs

select contractor

uses –
yields *contractor* : ContractorInfo

submit request

uses *documents* : CompensationDocs,
declaration : ContractorDecl
yields –

Basics

enter

uses –
yields *info* : *a*

parallel



view

uses *info* : *a*
yields –

check

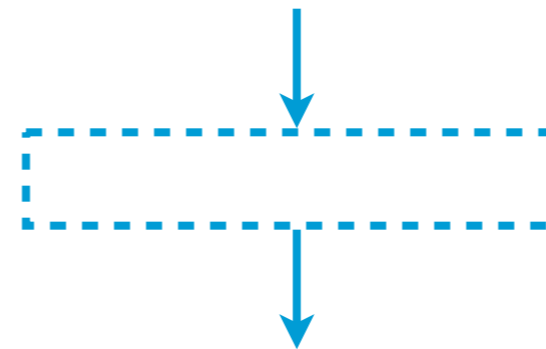


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-



Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	

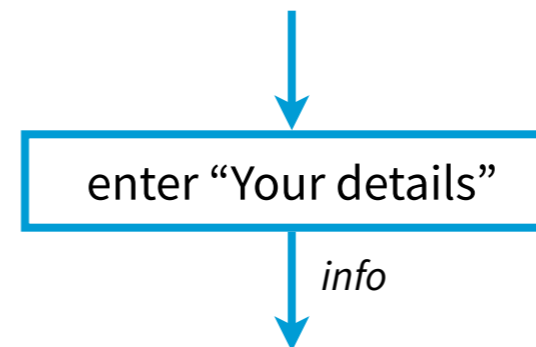


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration uses <i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo yields <i>declaration</i> : ContractorDecl	
provide documents uses – yields <i>documents</i> : CompensationDocs	
select contractor uses – yields <i>contractor</i> : ContractorInfo	
submit request uses <i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl yields –	

Basics	
enter uses – yields <i>info</i> : <i>a</i>	parallel 
view uses <i>info</i> : <i>a</i> yields –	check 

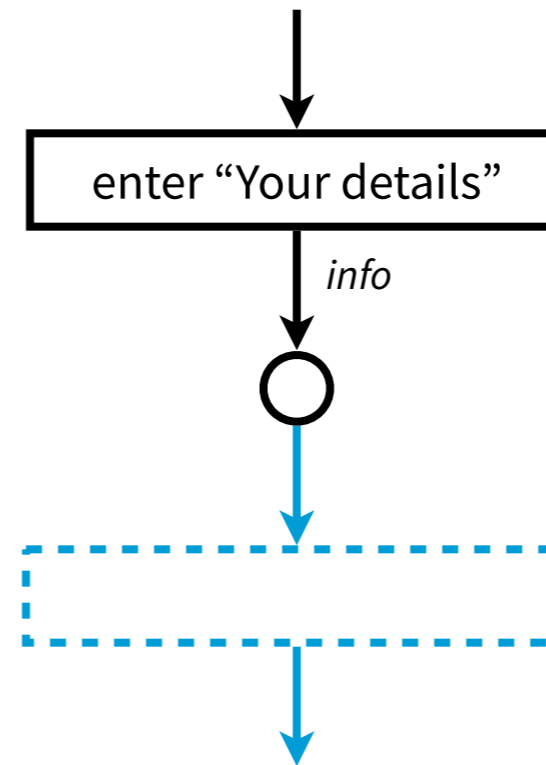


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses <i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo	
yields <i>declaration</i> : ContractorDecl	
provide documents	
uses -	
yields <i>documents</i> : CompensationDocs	
select contractor	
uses -	
yields <i>contractor</i> : ContractorInfo	
submit request	
uses <i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl	
yields -	

Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	

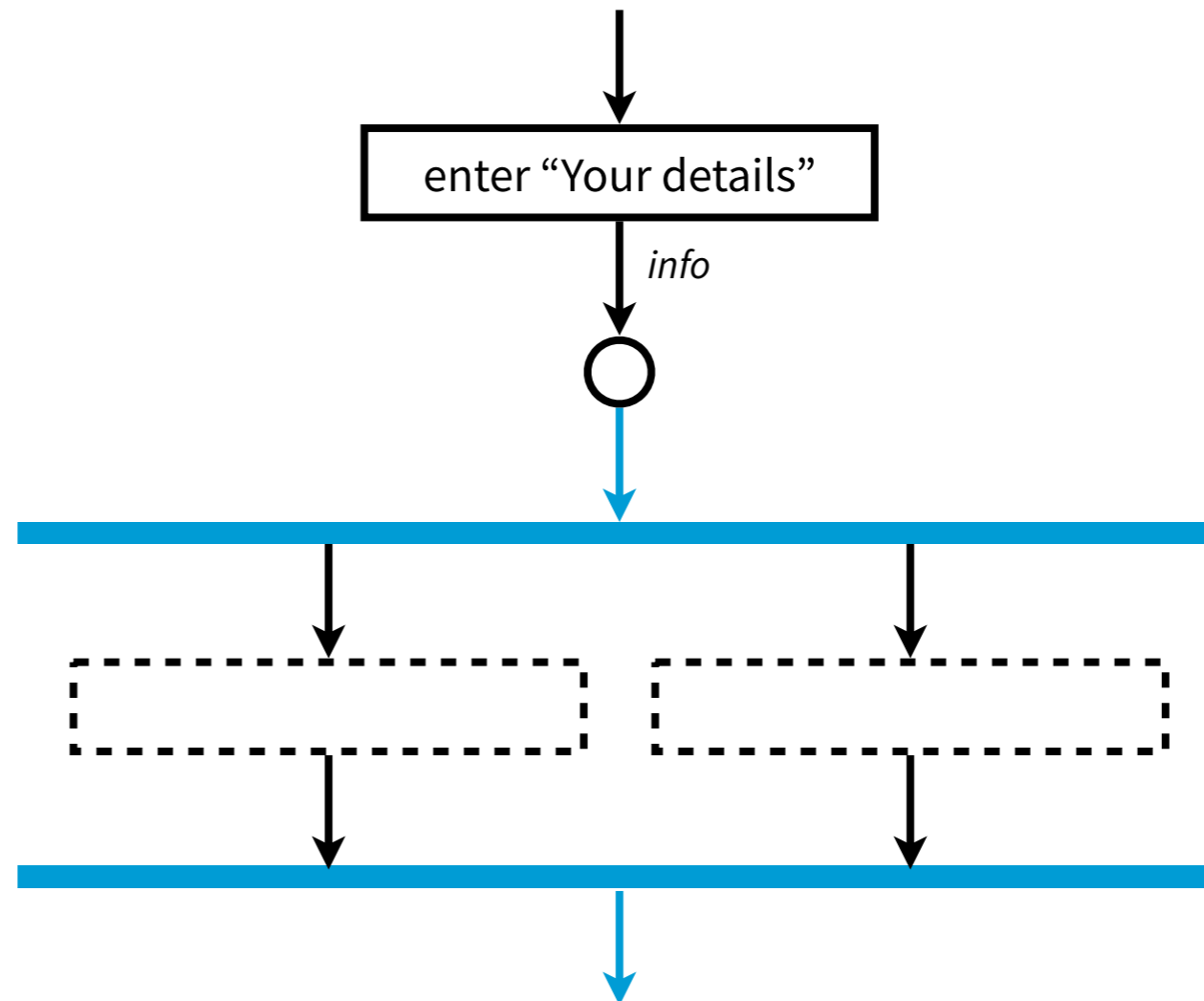


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

Basics	
enter	parallel
uses -	▬
yields <i>info</i> : a	
view	check
uses <i>info</i> : a	◆
yields -	

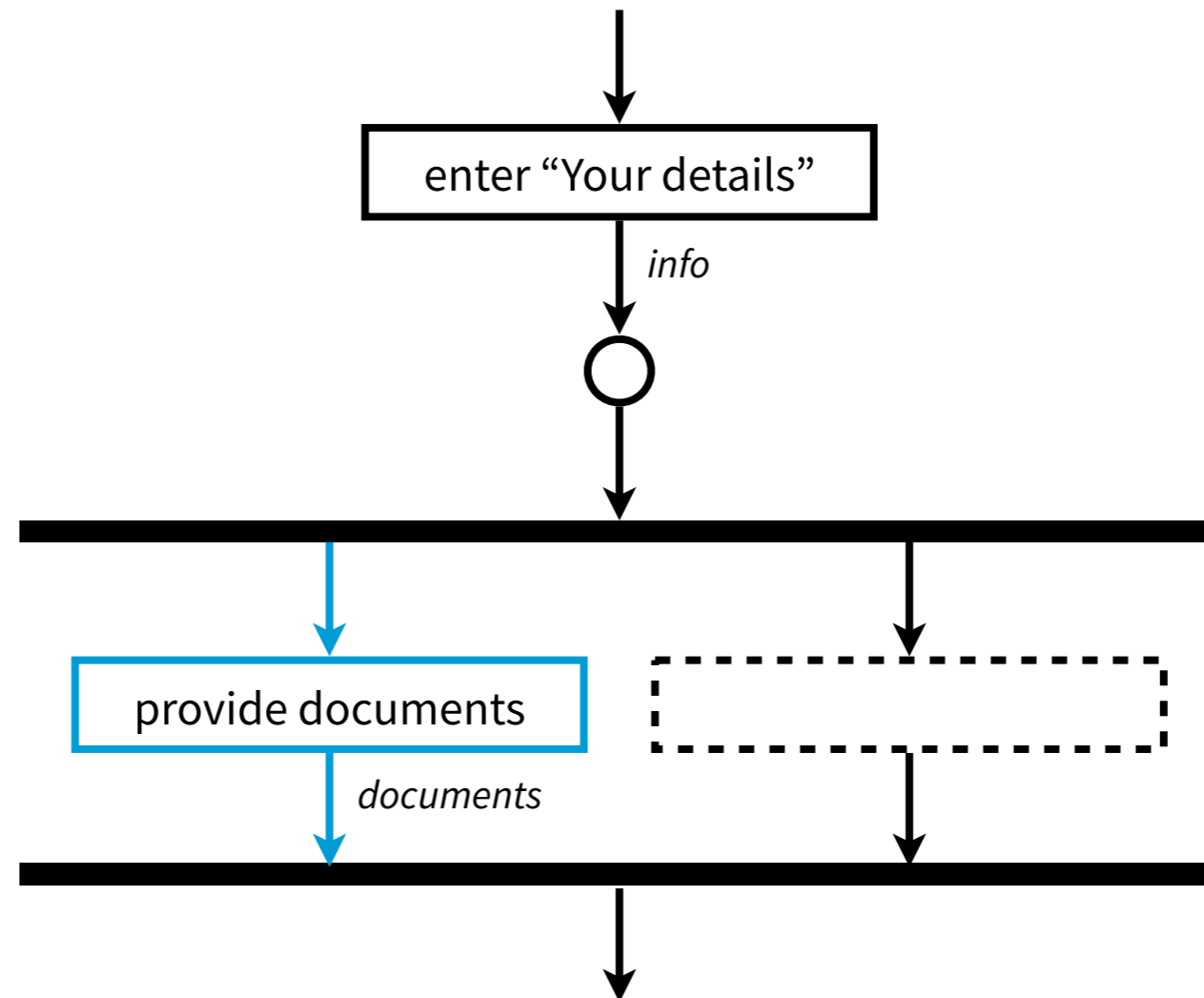


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

Basics	
enter	parallel
uses -	▬
yields <i>info</i> : a	
view	check
uses <i>info</i> : a	◆
yields -	

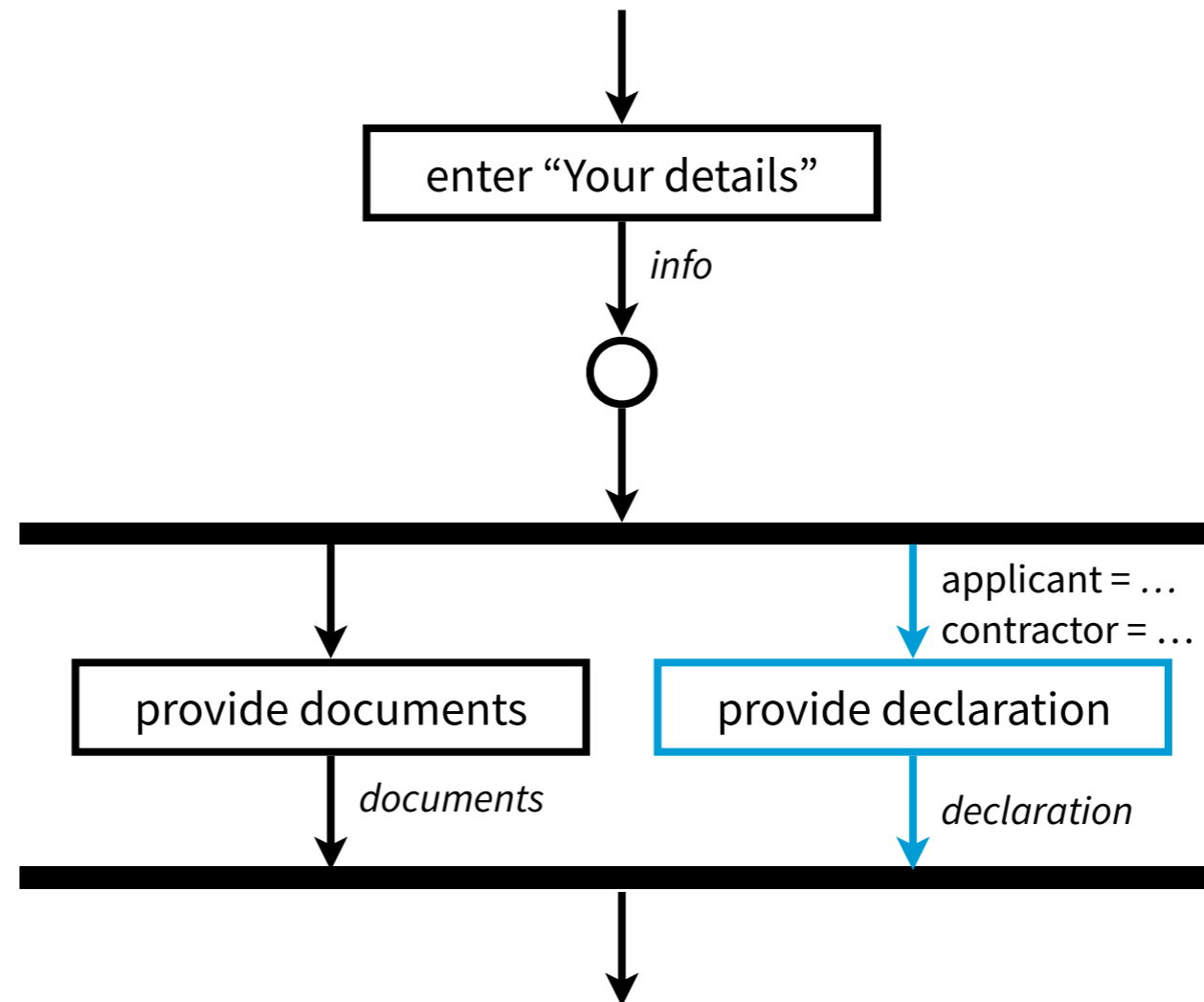


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	

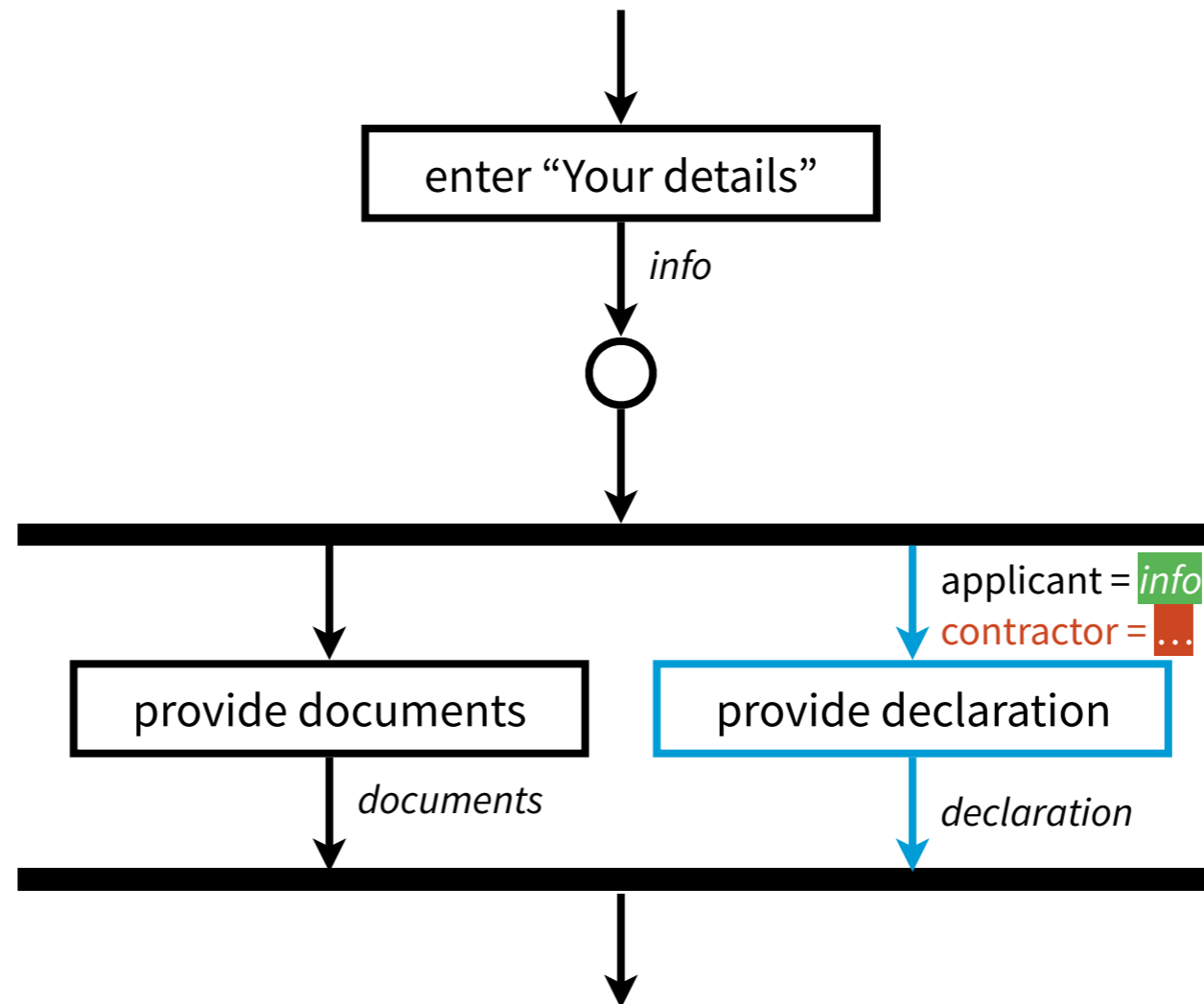


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	



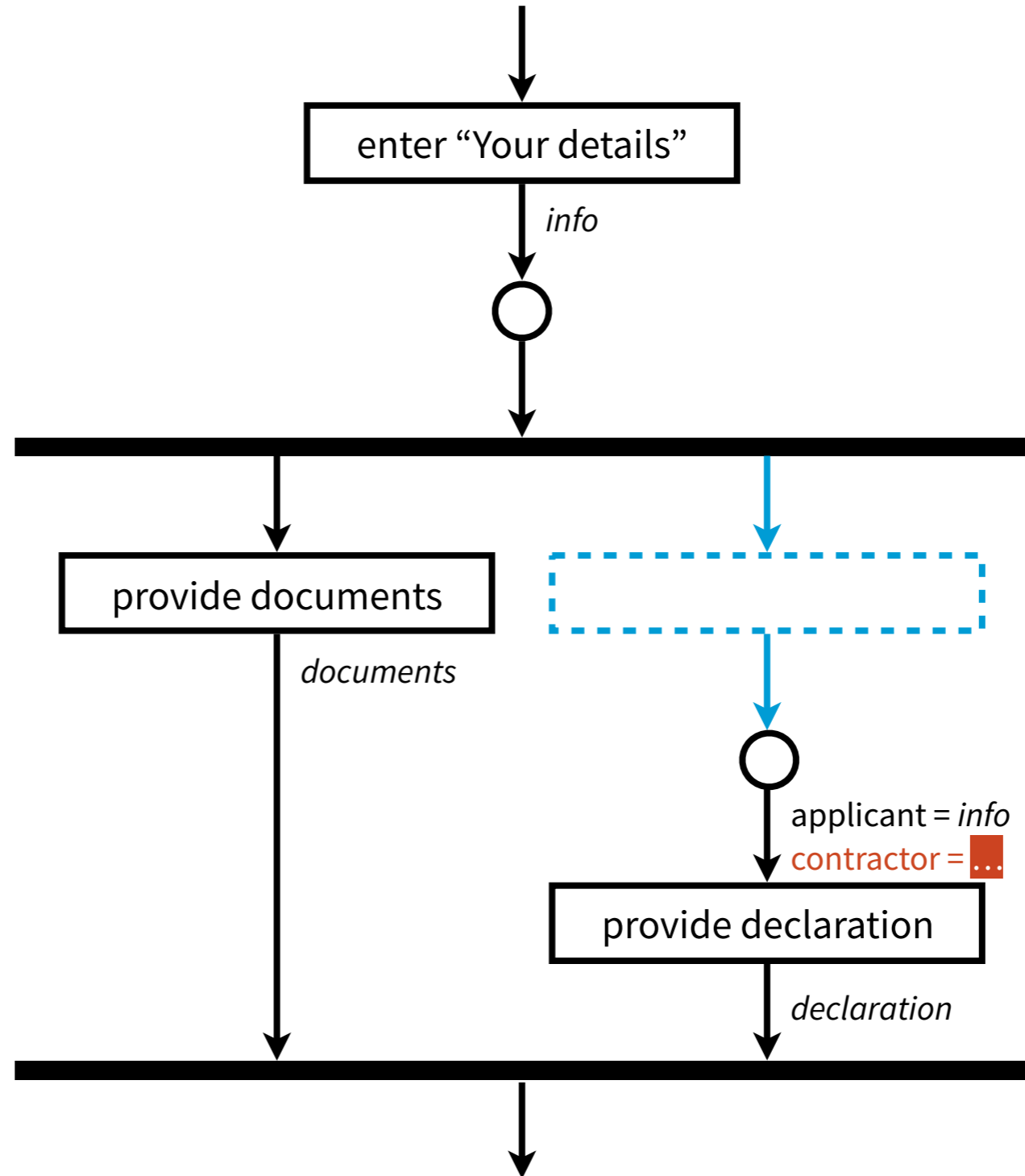
← Error

Nothing of type
'ContractorInfo'
in scope, maybe
produce one
before?



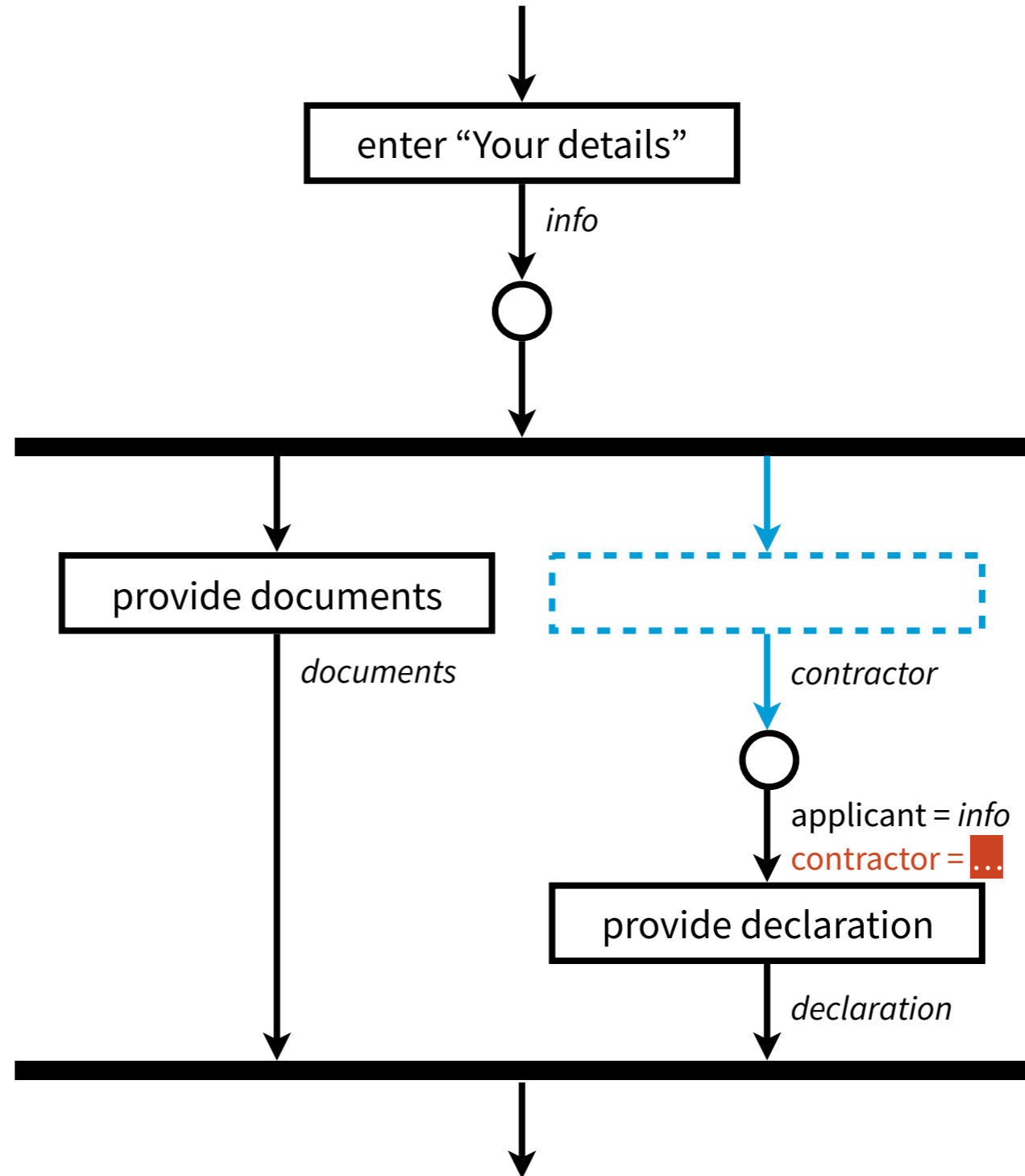
request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift
Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-
Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	



request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift
Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-
Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	

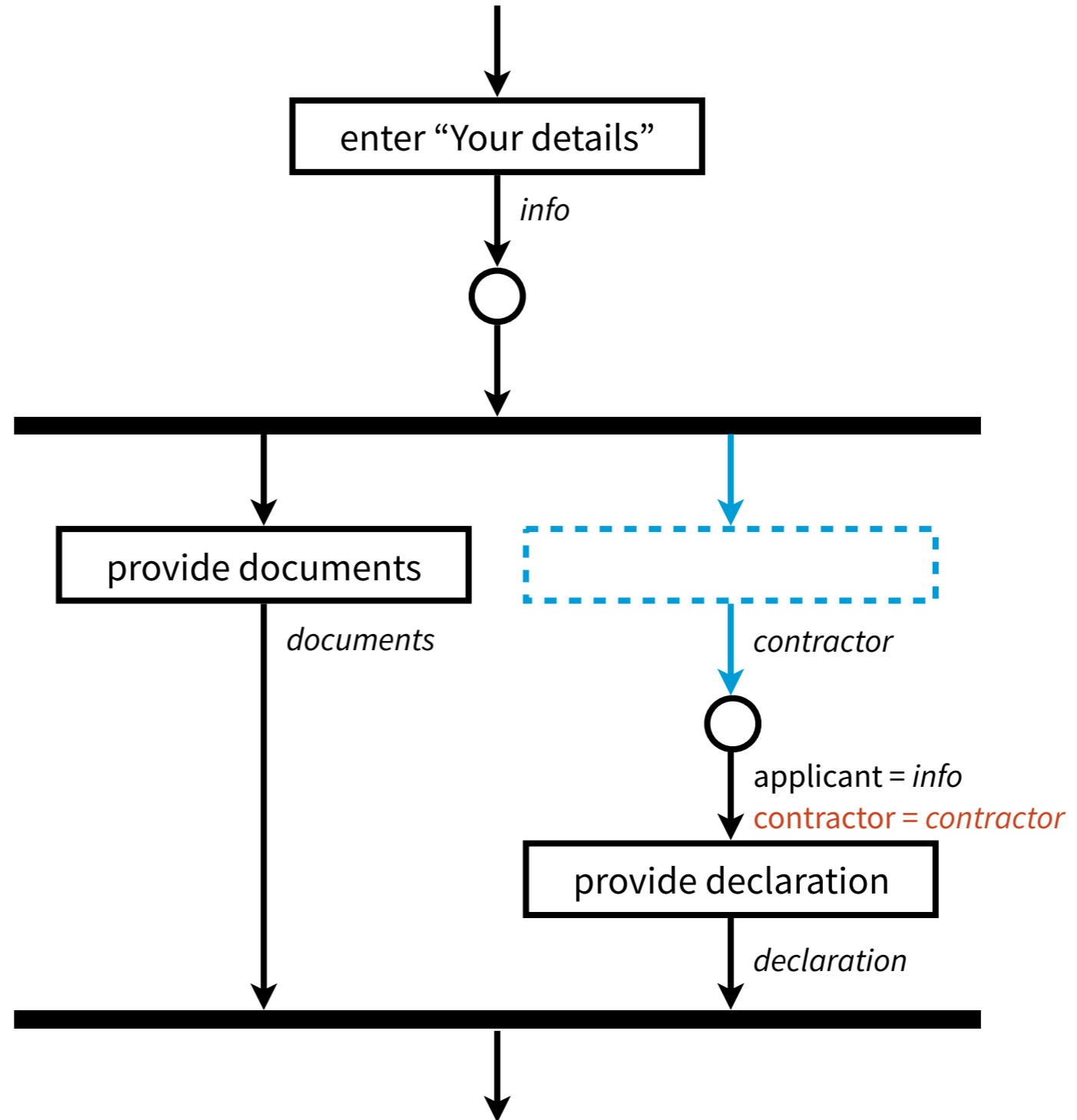


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

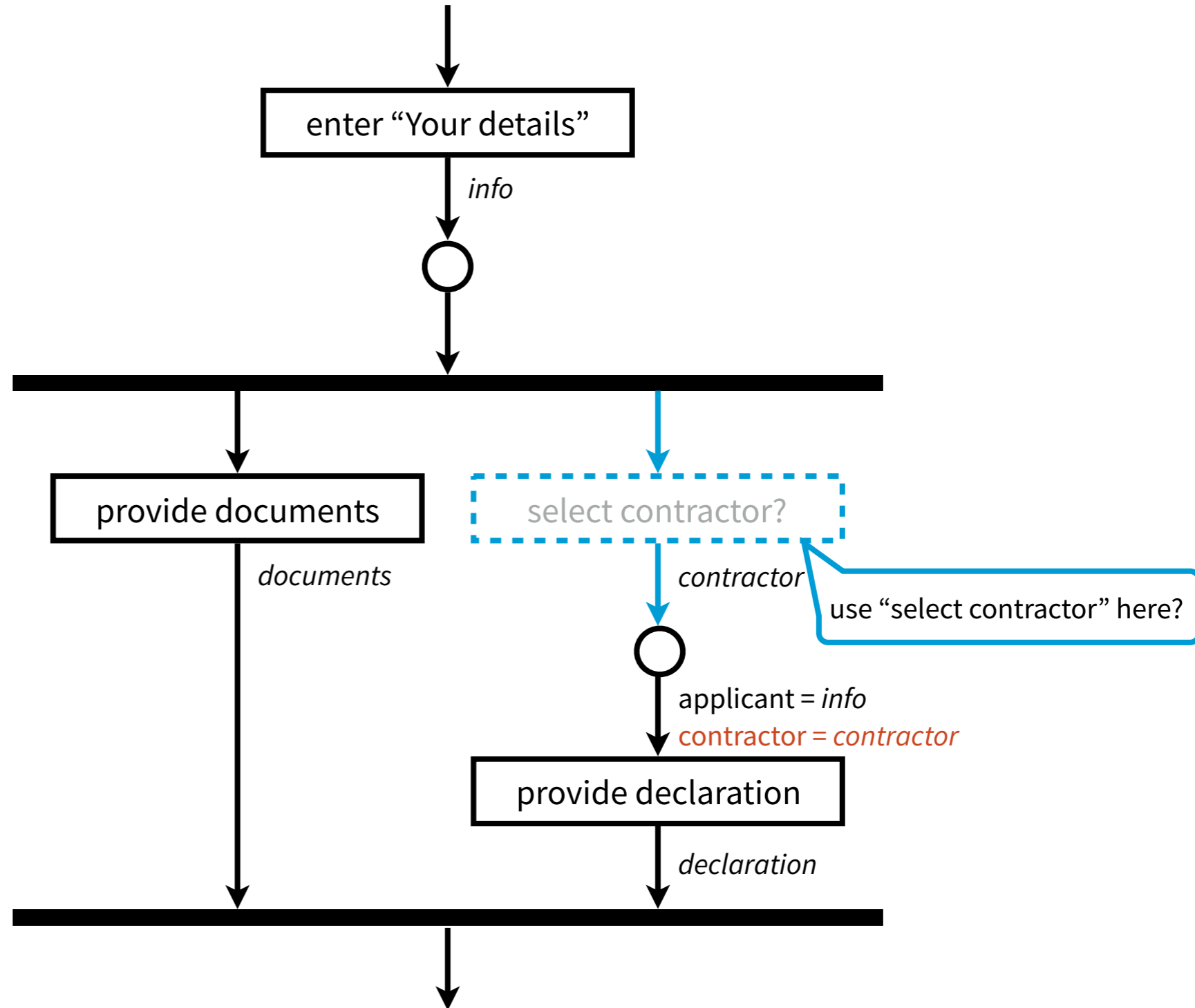
Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	



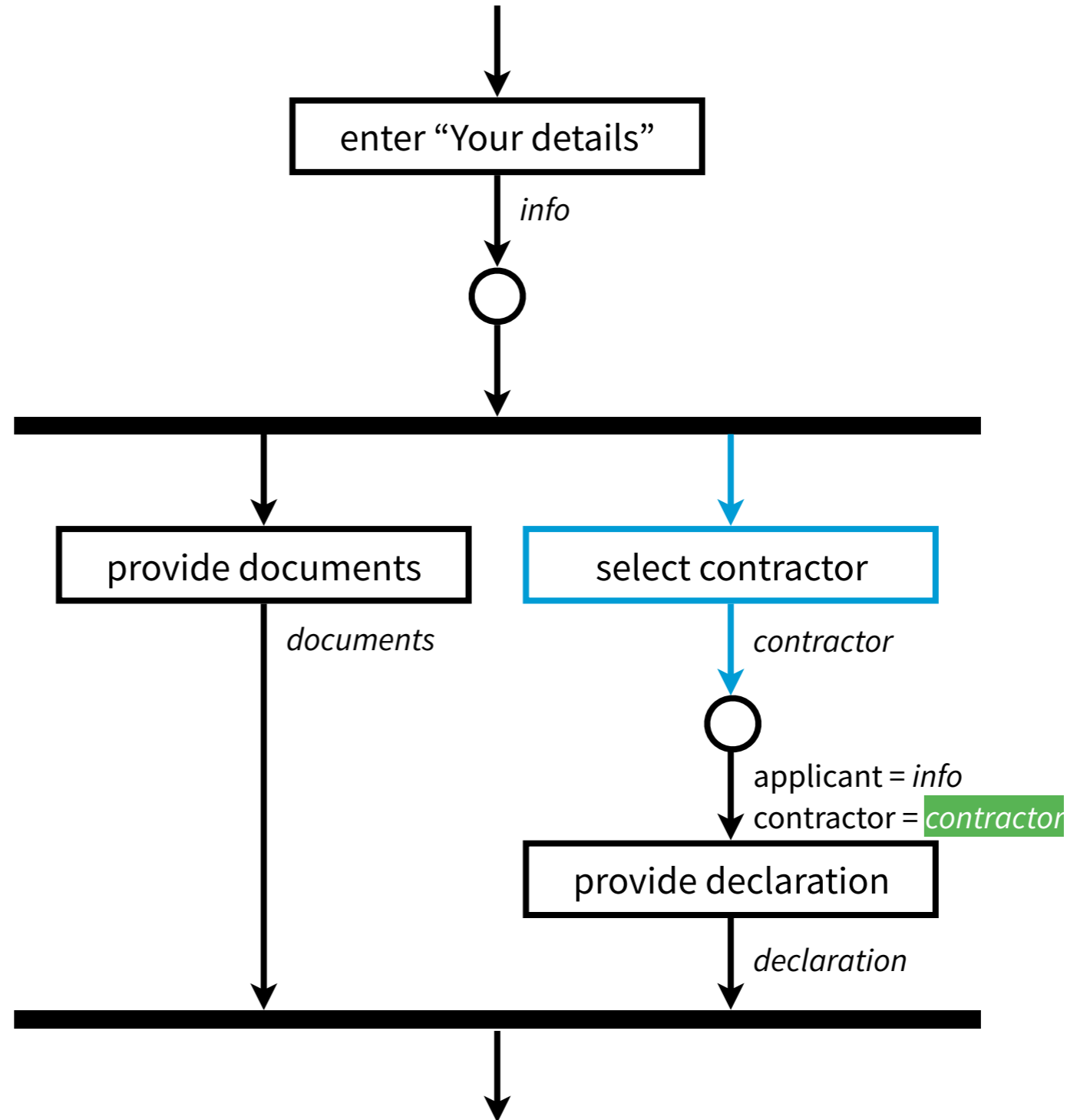
request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift
Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-
Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	



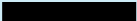

request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift
Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-
Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	

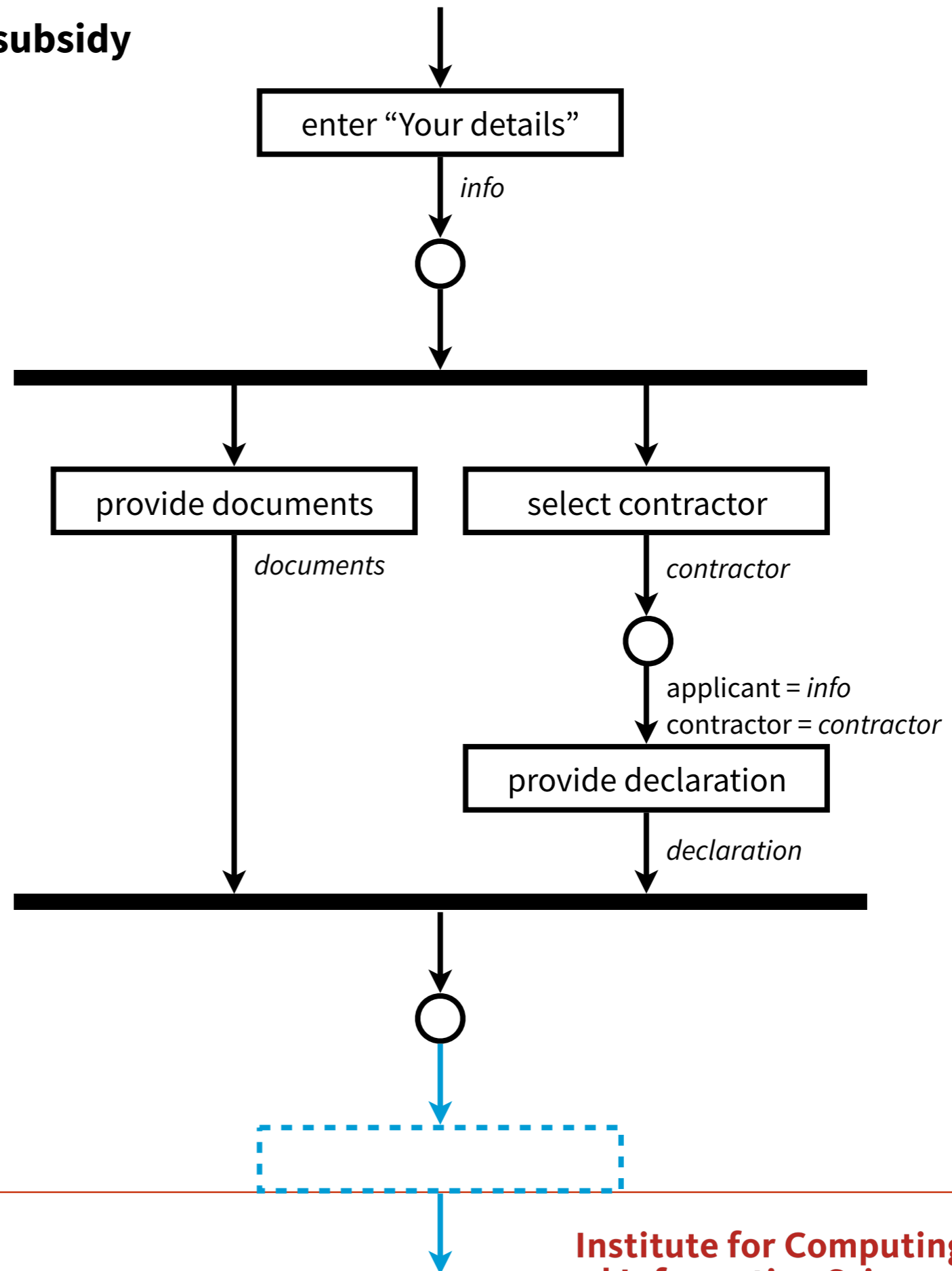


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-



Basics	
enter	parallel
uses -	
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	
yields -	

request subsidy

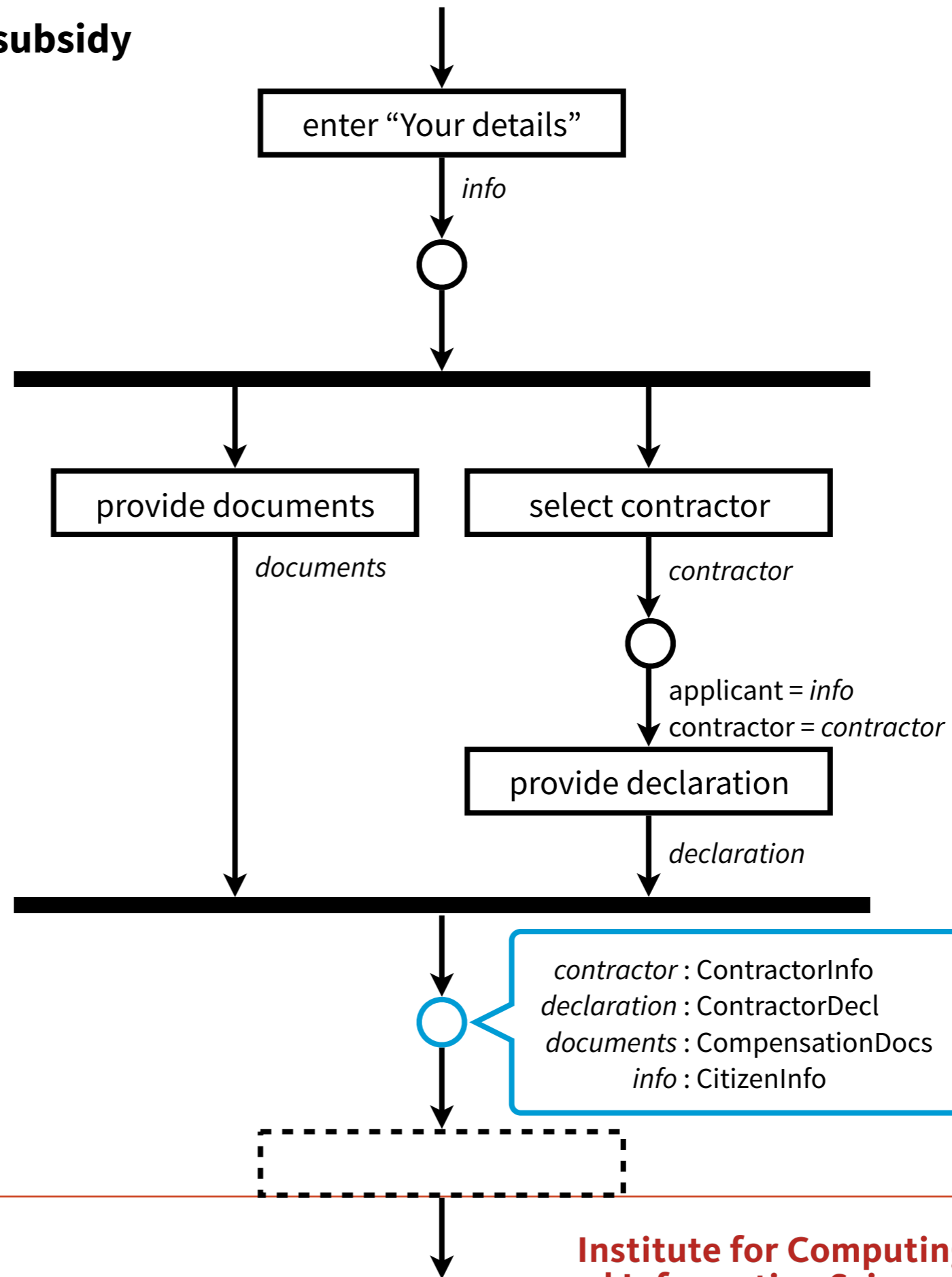


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

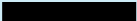

Basics	
enter	parallel
uses -	
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	
yields -	

request subsidy

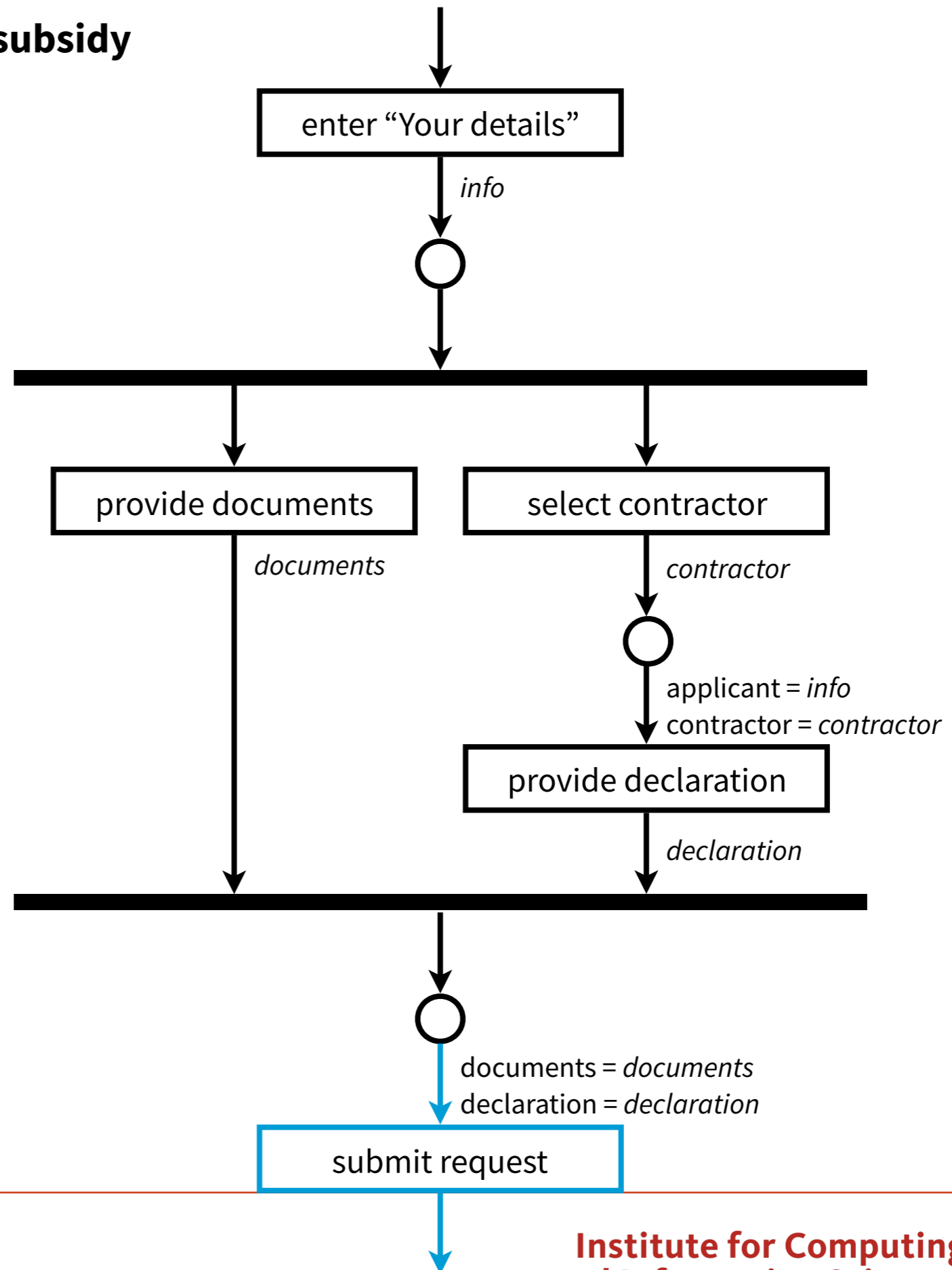


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

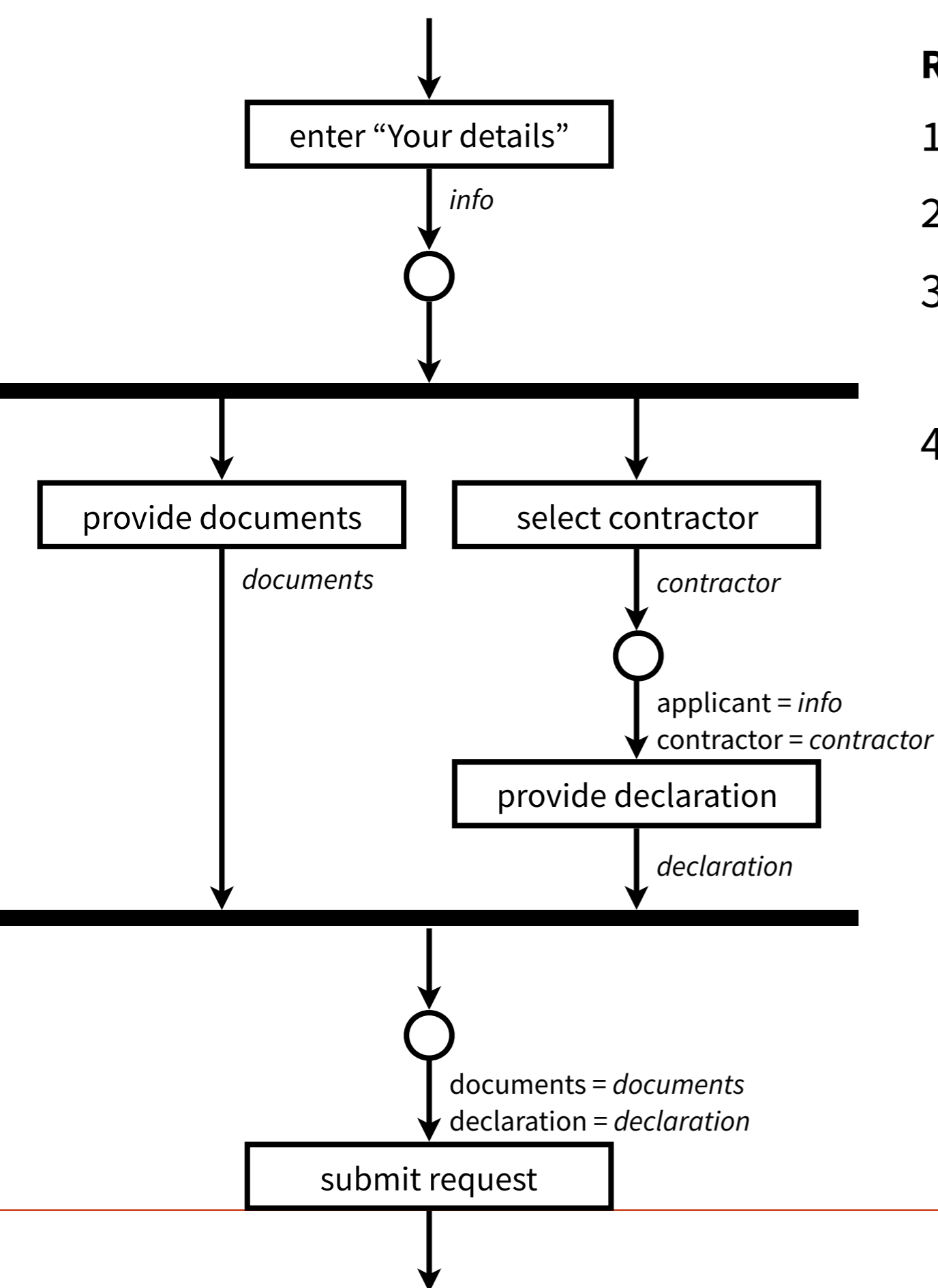
Basics	
enter	parallel
uses -	
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	
yields -	

request subsidy



Request solar panel subsidy

1. present a webform to an applicant
2. check if he/she is obligable for a subsidy
3. ask for tax compensation documents and a contractor declaration
4. submit the request to the tax office



enter "Your details"

info



provide documents

documents

select contractor

contractor



applicant = *info*
contractor = *contractor*

provide declaration

declaration



documents = *documents*
declaration = *declaration*

submit request

1. present a webform to an applicant
2. check if he/she is obligable for a subsidy
3. ask for tax compensation documents
and a contractor declaration
4. submit the request to the tax office

enter "Your details"

info



provide documents

documents

select contractor

contractor



applicant = info
contractor = contractor

provide declaration

declaration



documents = documents
declaration = declaration

submit request

1. present a webform to an applicant
2. check if he/she is obligable for a subsidy
3. ask for tax compensation documents
and a contractor declaration
4. submit the request to the tax office

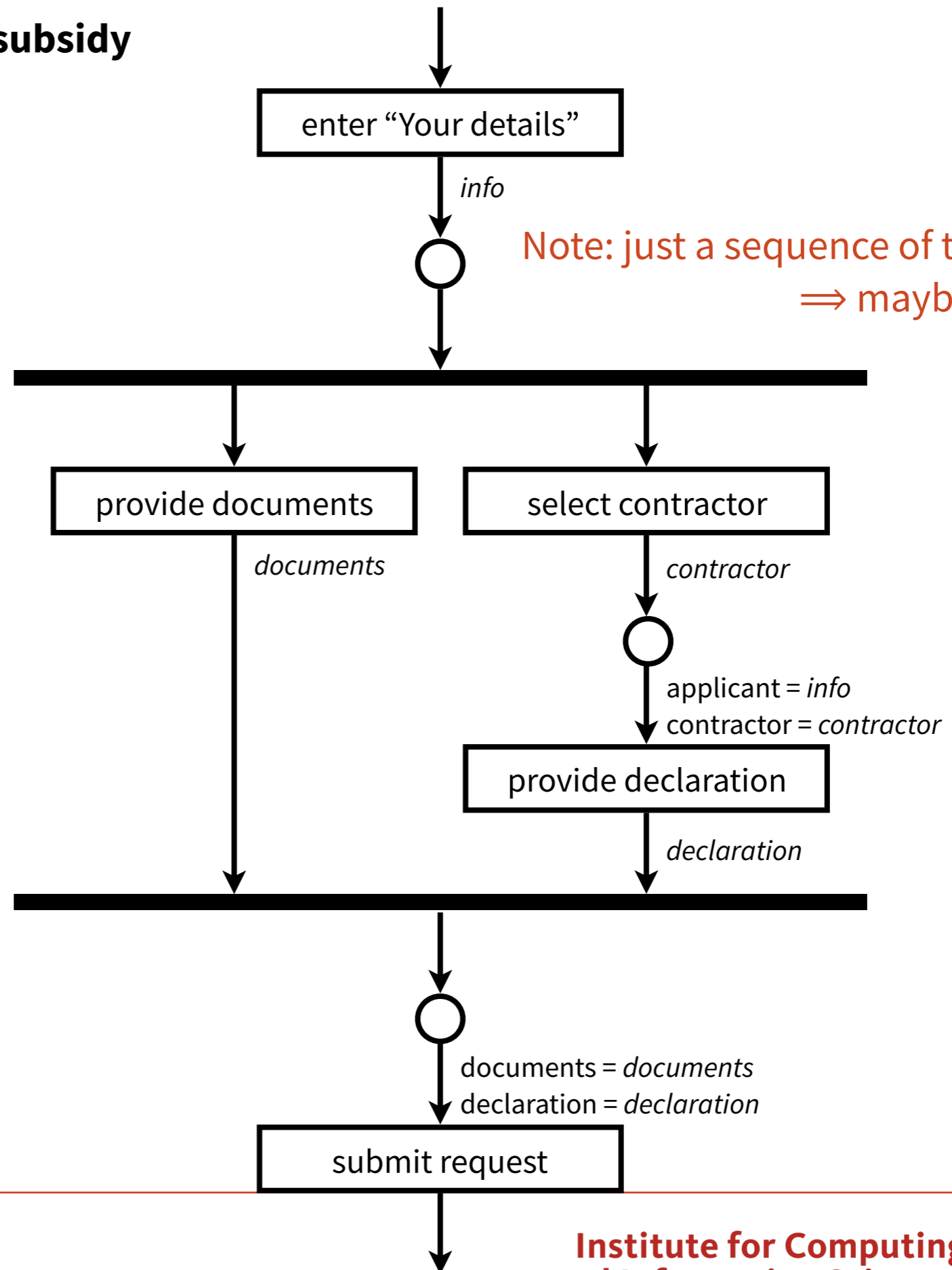


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-



Basics	
enter	parallel
uses -	▬
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	◆
yields -	

request subsidy

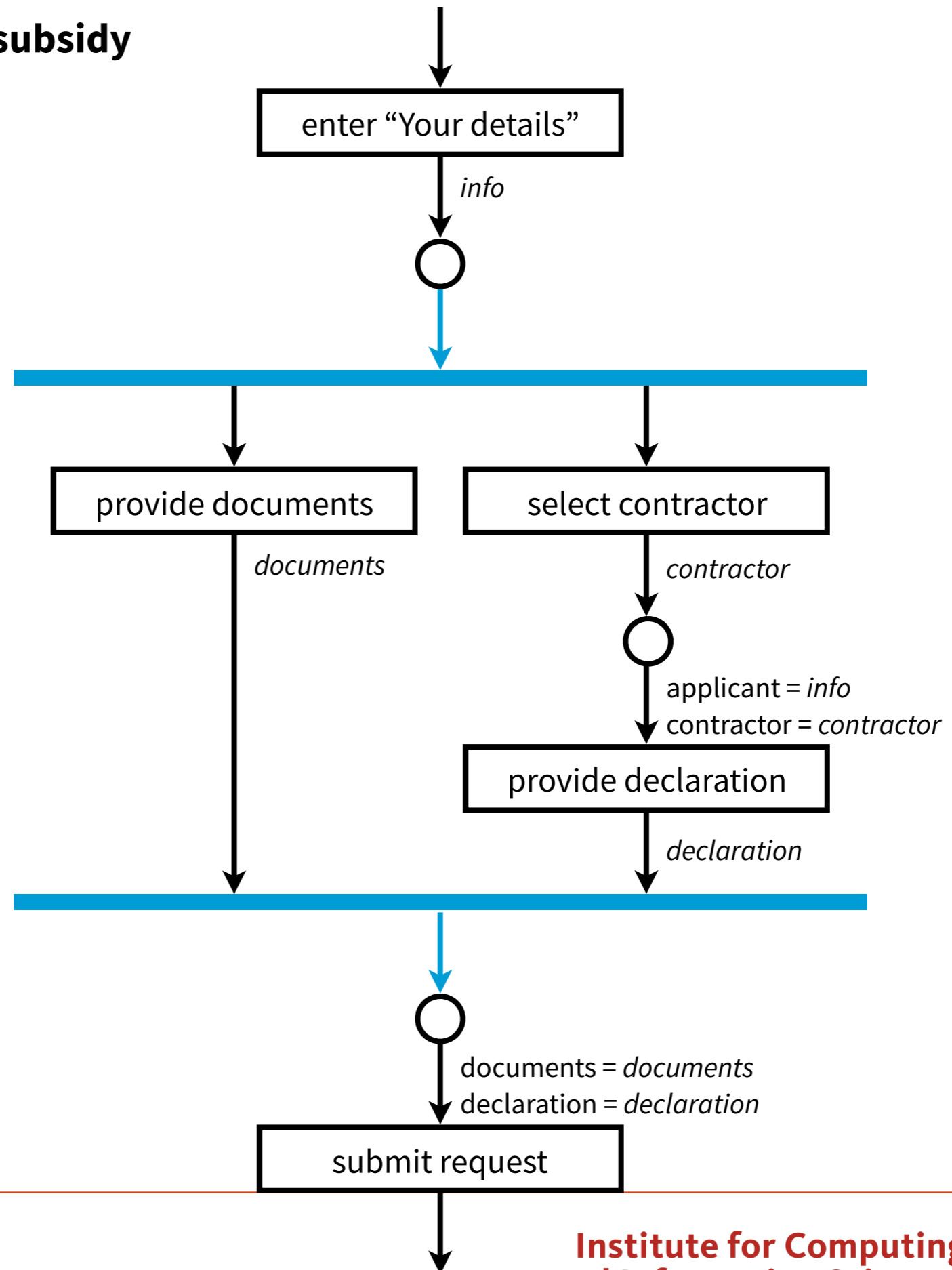


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-



Basics	
enter	parallel
uses -	
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	
yields -	

request subsidy

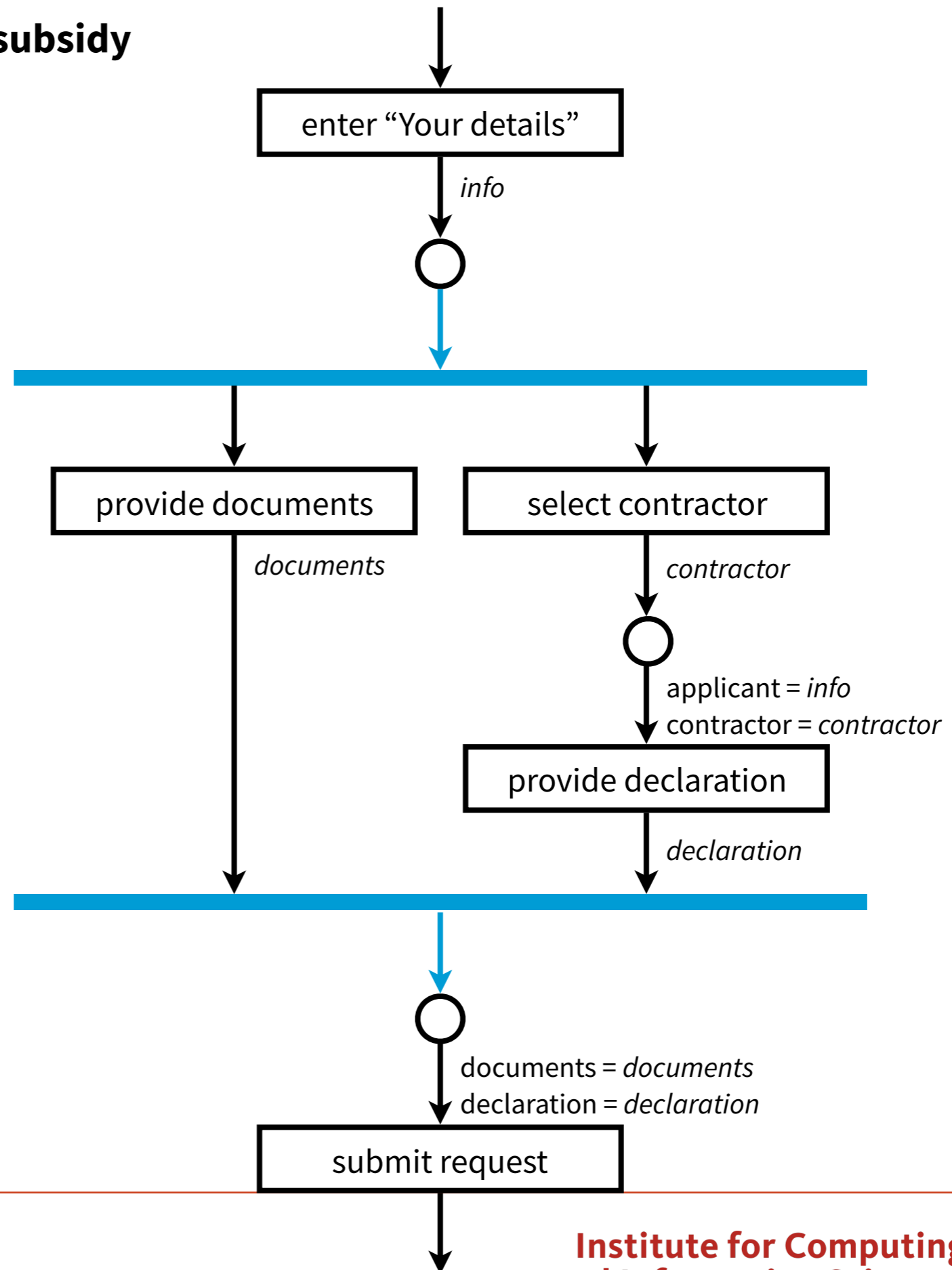


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
provide declaration	
uses	<i>applicant</i> : CitizenInfo, <i>contractor</i> : ContractorInfo
yields	<i>declaration</i> : ContractorDecl
provide documents	
uses	-
yields	<i>documents</i> : CompensationDocs
select contractor	
uses	-
yields	<i>contractor</i> : ContractorInfo
submit request	
uses	<i>documents</i> : CompensationDocs, <i>declaration</i> : ContractorDecl
yields	-

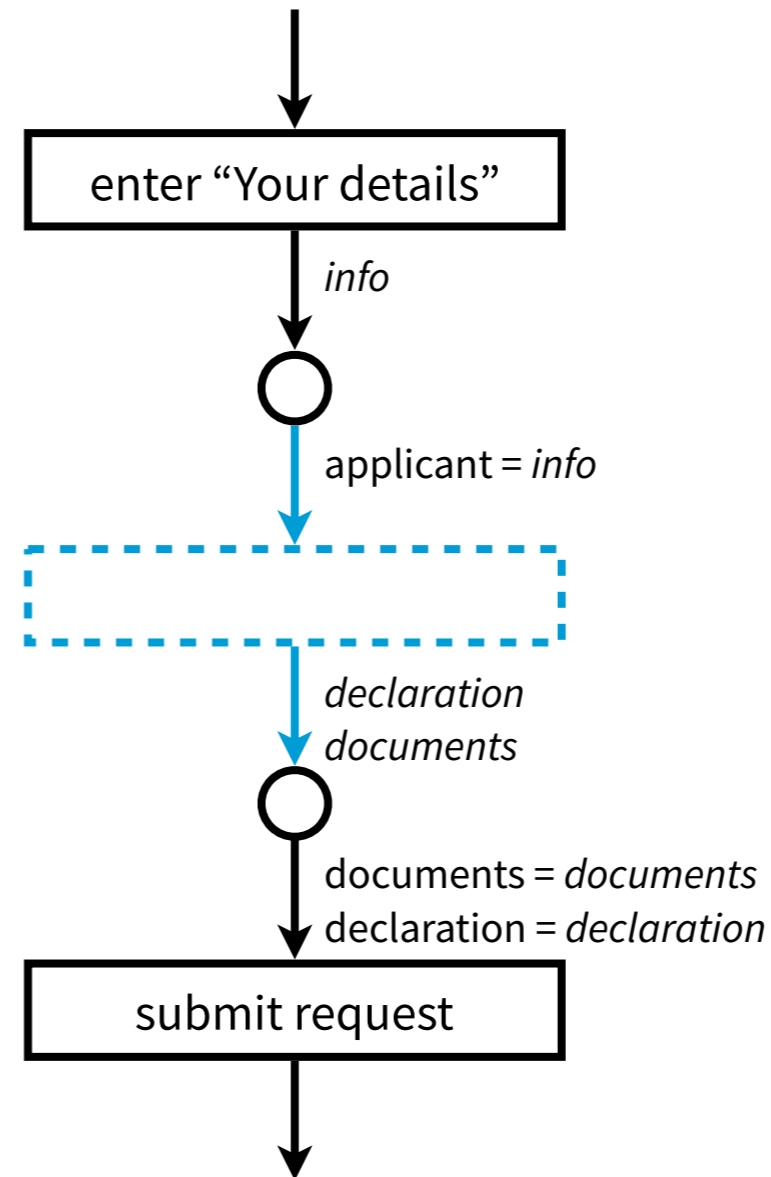
Basics	
enter	parallel
uses -	
yields <i>info</i> : <i>a</i>	
view	check
uses <i>info</i> : <i>a</i>	
yields -	

request subsidy



request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift
Library	
... <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	
Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

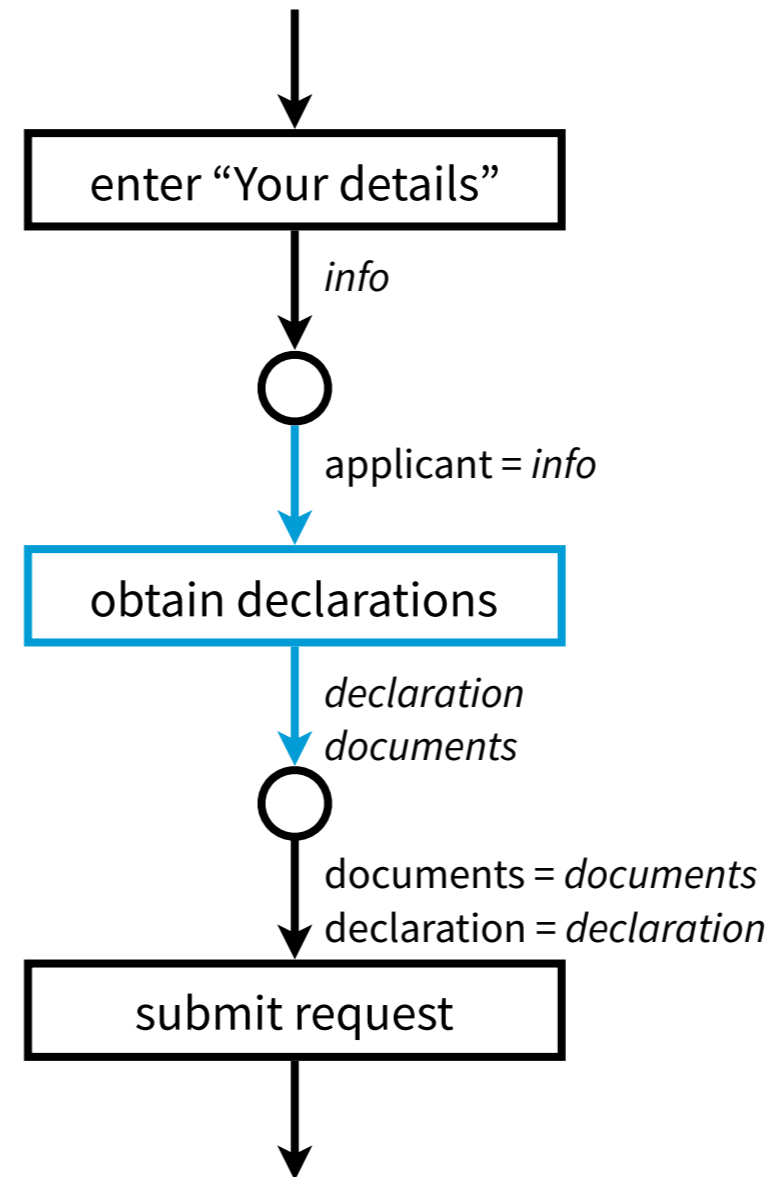


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

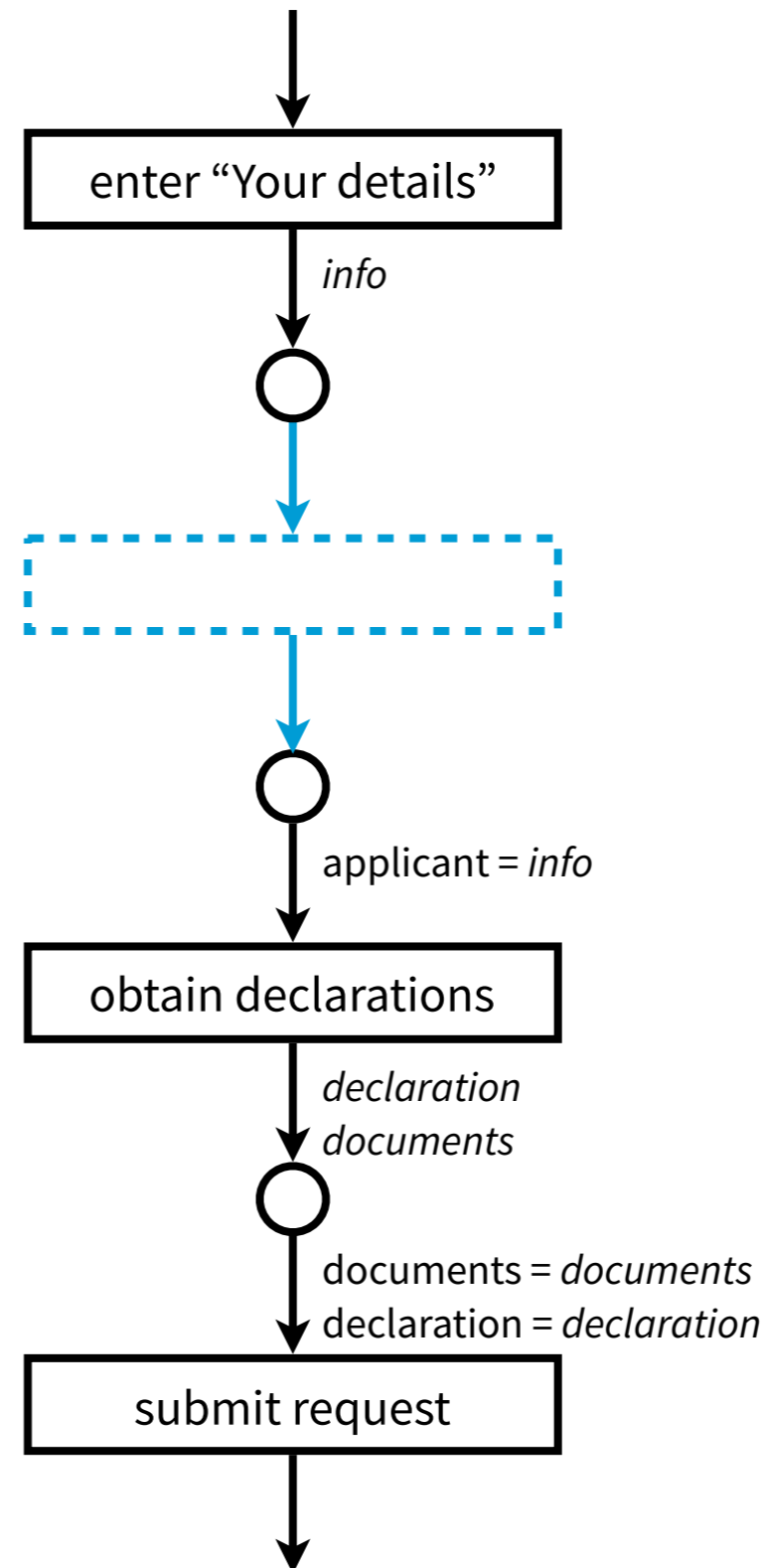


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

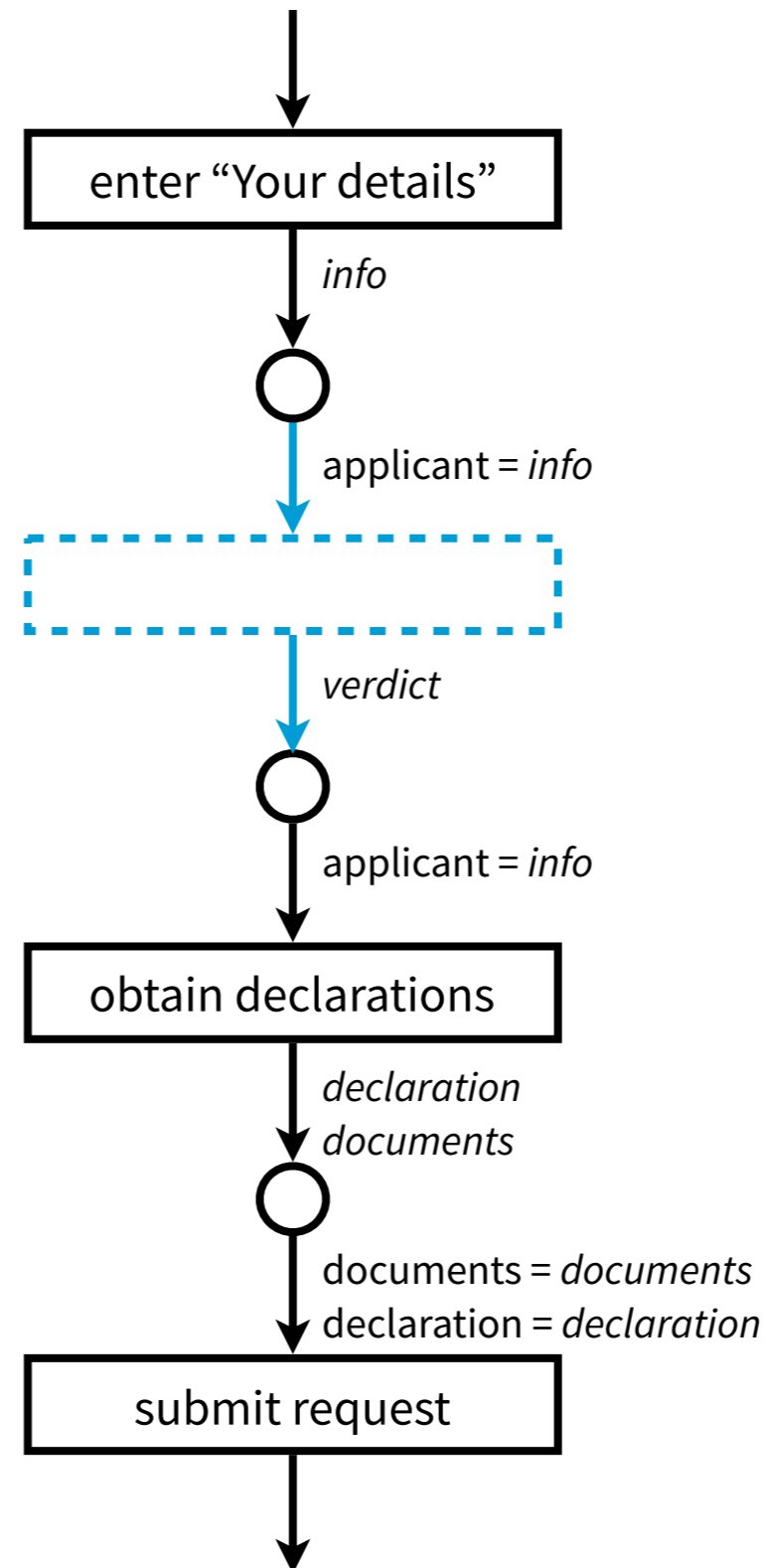


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

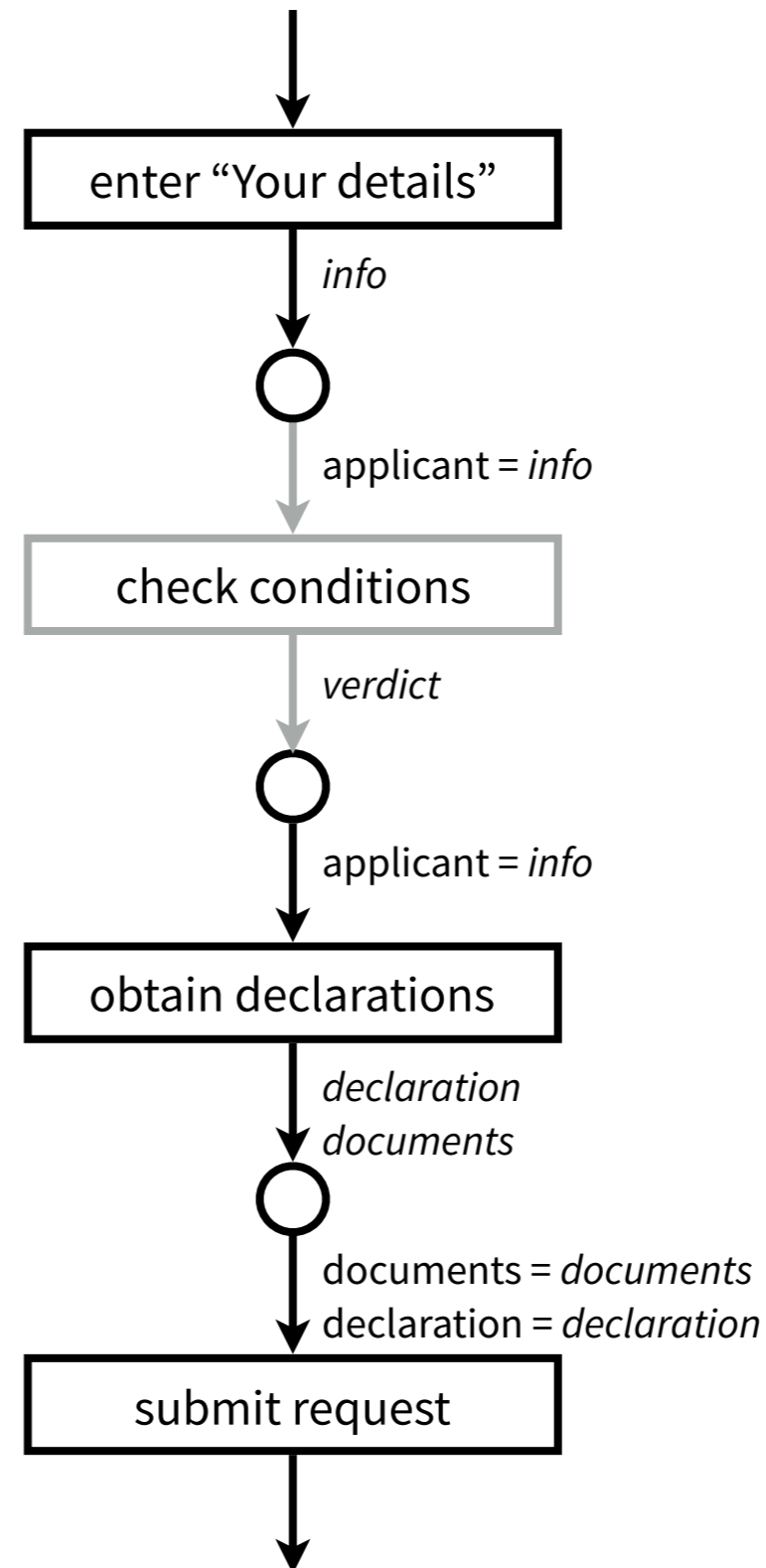
Library	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info : a</i>	parallel ▬
view uses <i>info : a</i> yields -	check ◆



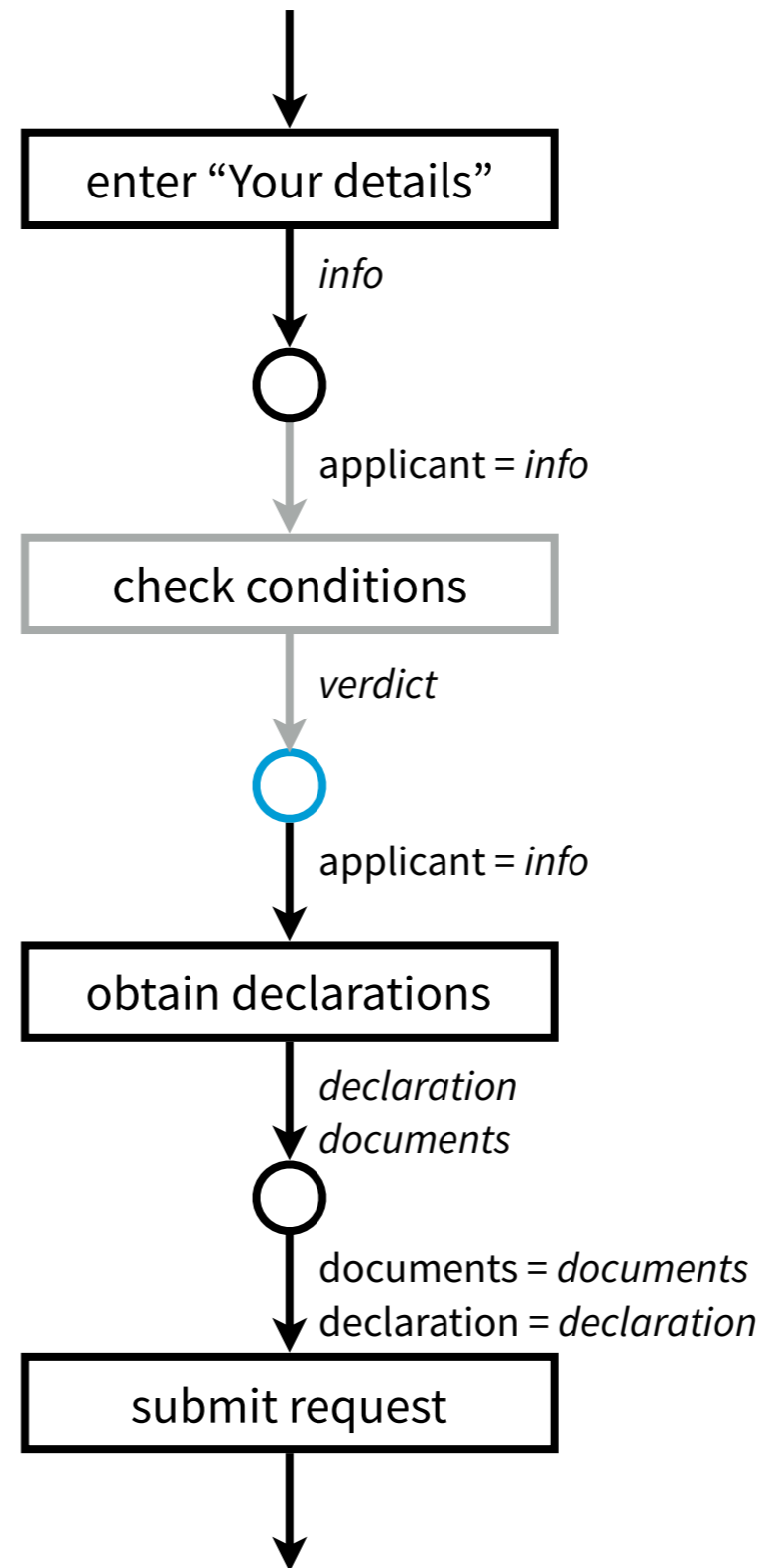
request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift
Library	
check conditions	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	
Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆



request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift
Library	
check conditions	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	
Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

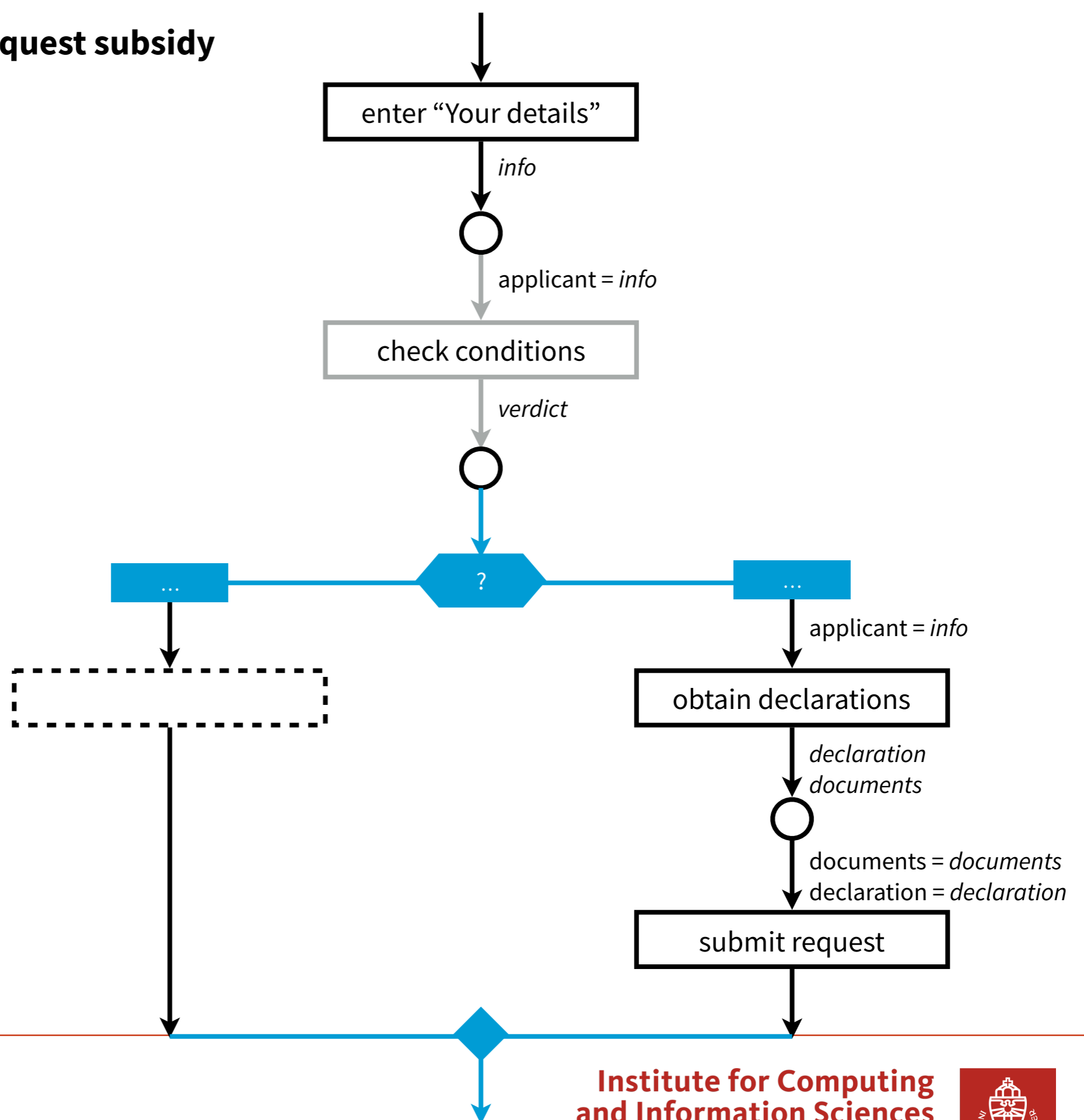


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
check conditions	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

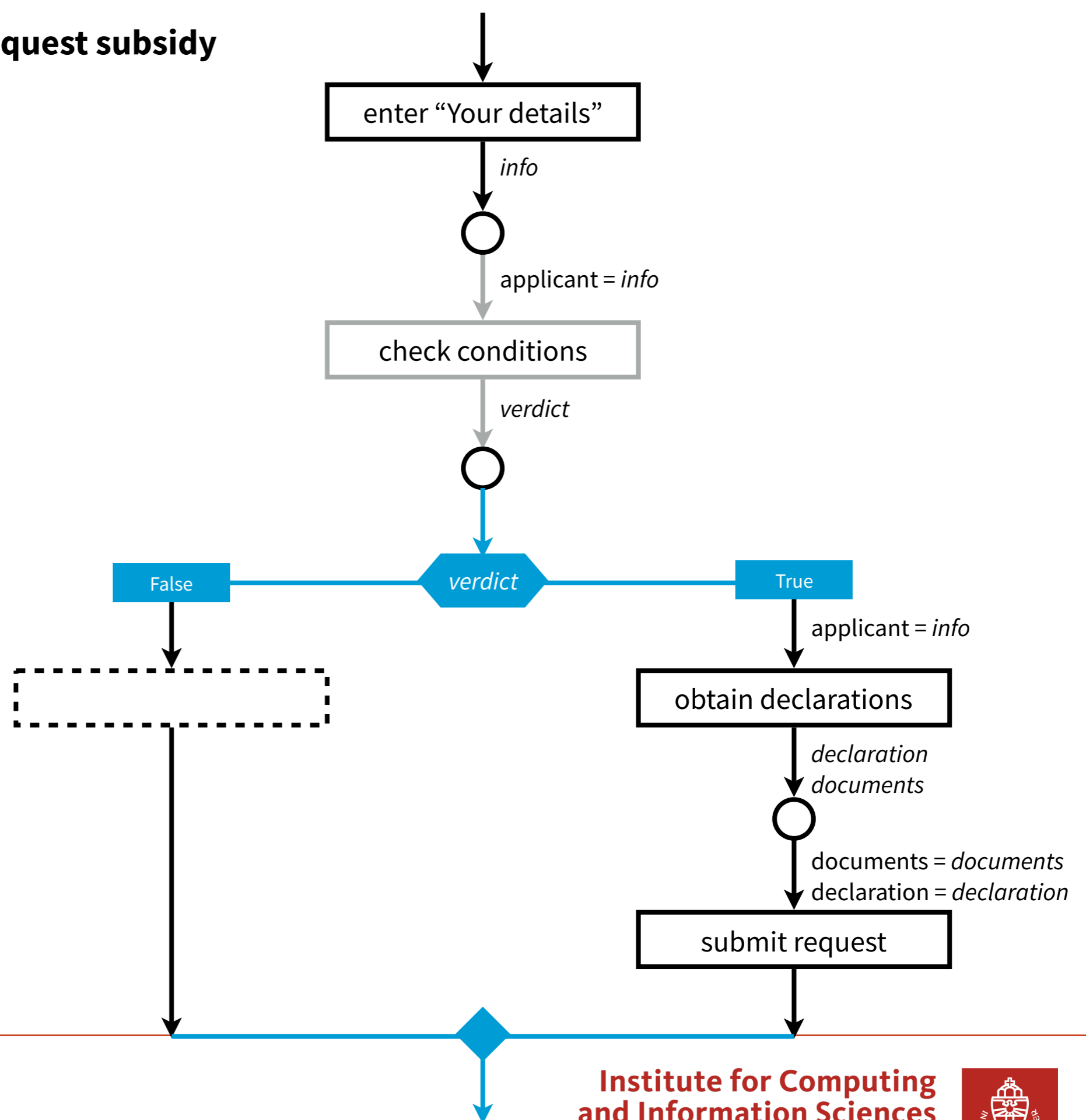


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
check conditions	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

request subsidy

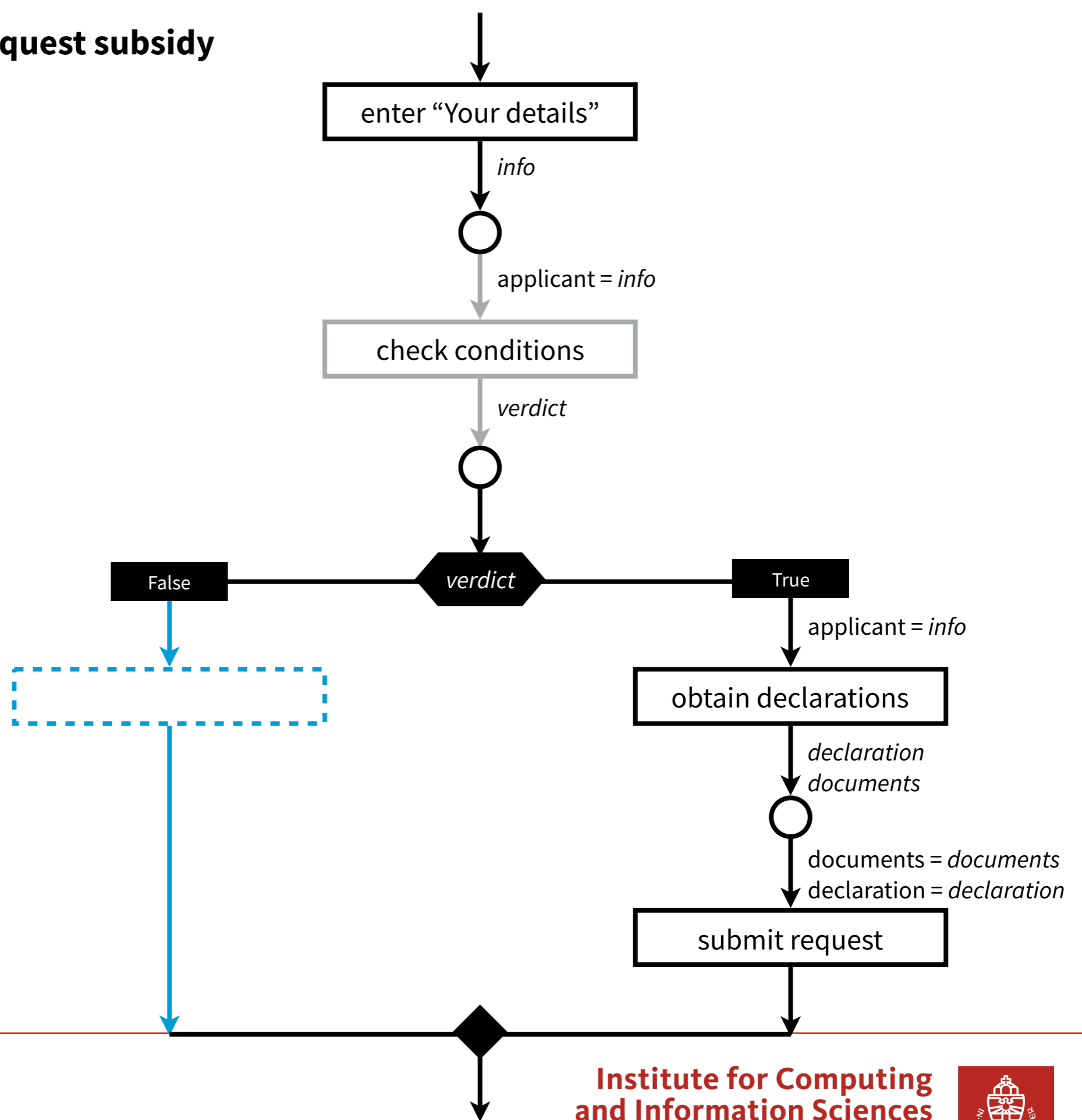


request subsidy

Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
check conditions	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

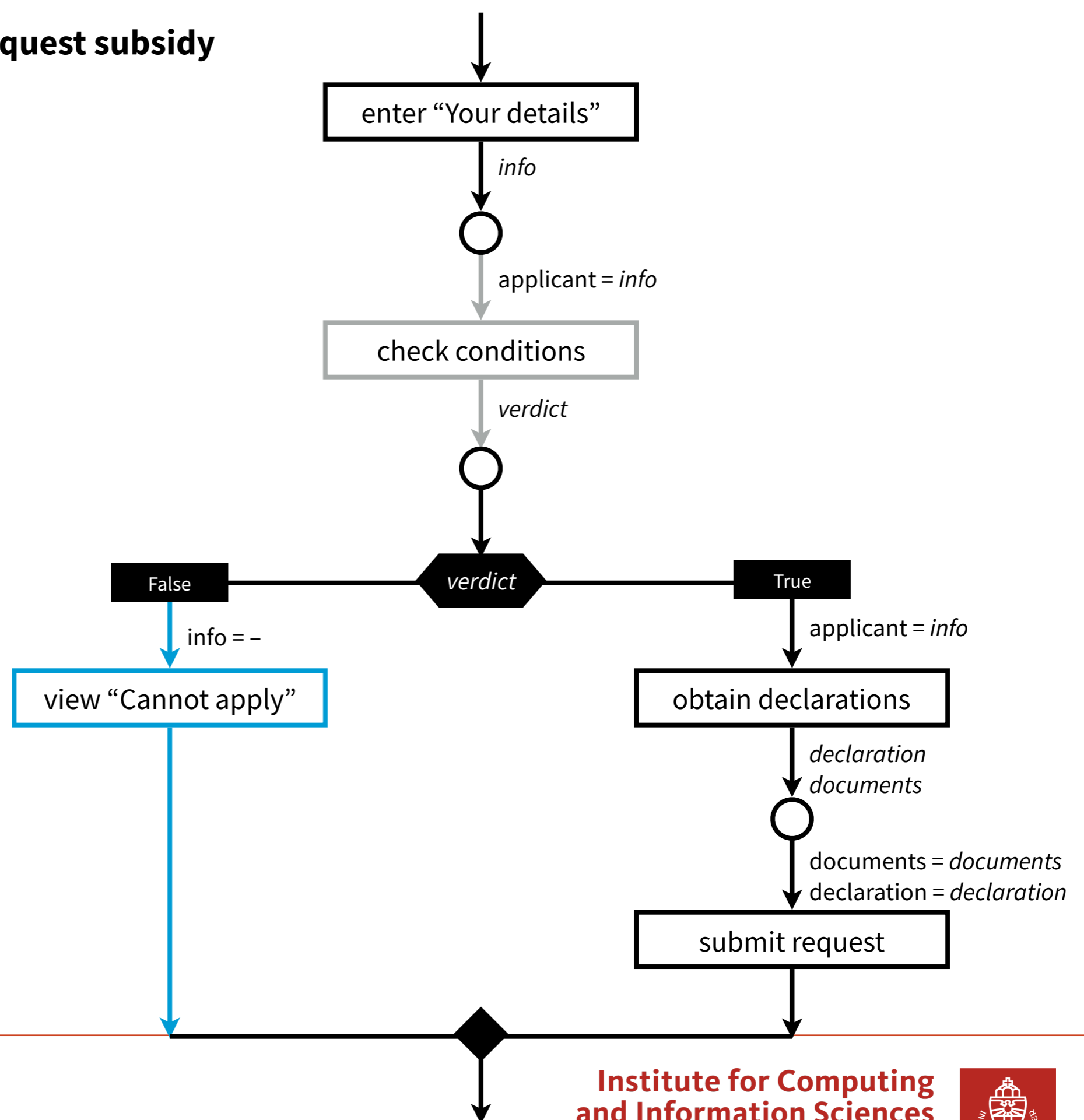


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
check conditions	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

request subsidy

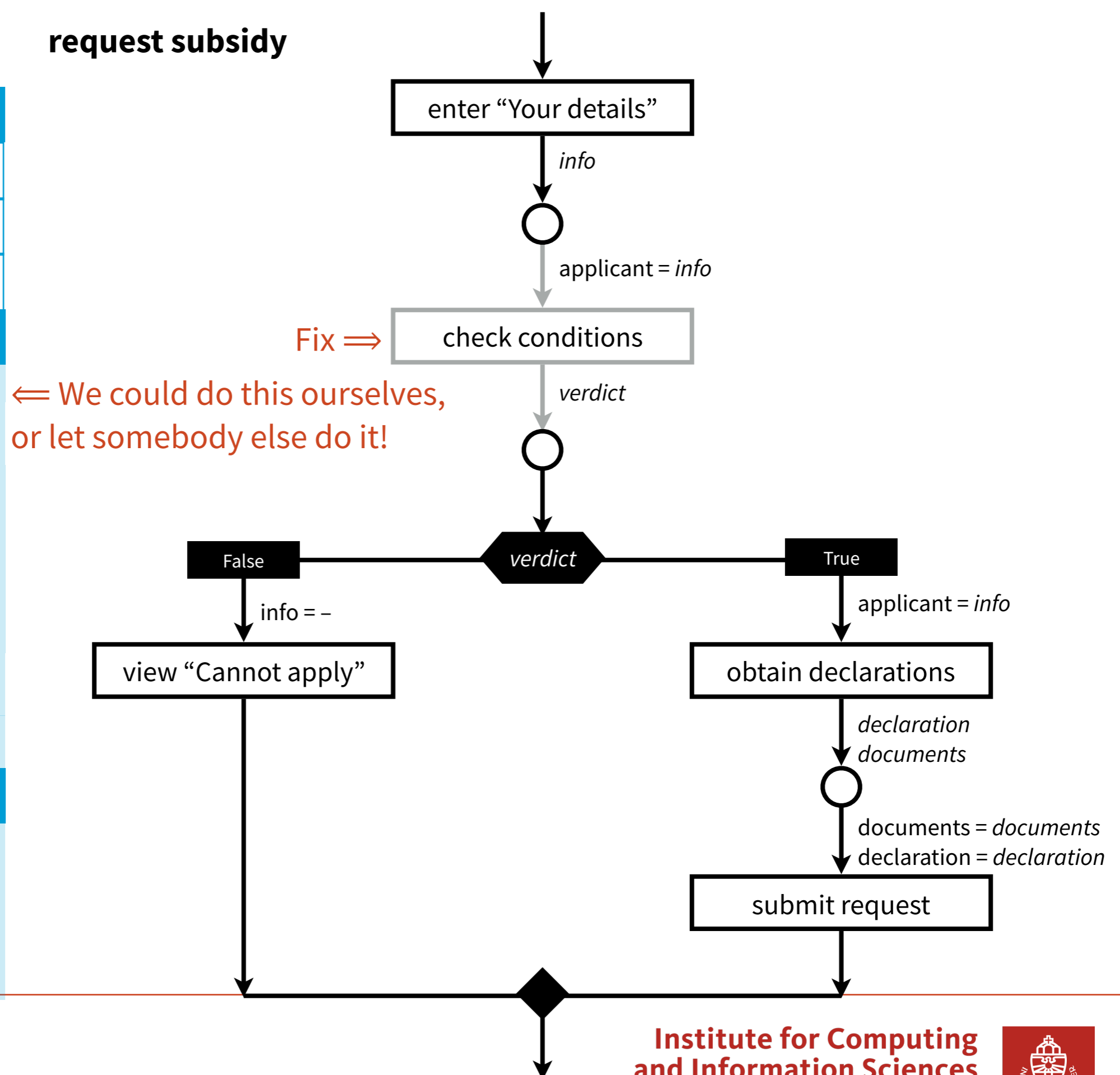


Operations	
Init	Extend ↑
Swap	Extend ↓
Split	Lift

Library	
<i>check conditions</i>	
obtain declarations <i>applicant</i> → <i>declaration, documents</i>	
provide declaration <i>applicant, contractor</i> → <i>declaration</i>	
provide documents - → <i>documents</i>	
select contractor - → <i>contractor</i>	
submit request <i>documents, declaration</i> → -	

Basics	
enter uses - yields <i>info: a</i>	parallel ▬
view uses <i>info: a</i> yields -	check ◆

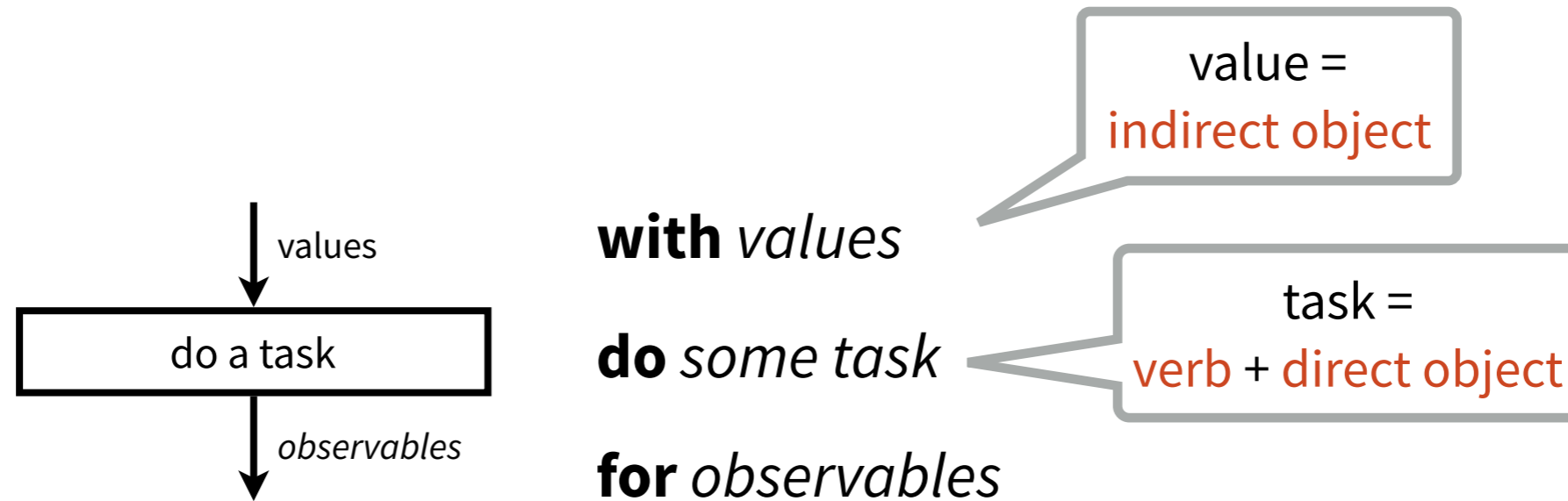
request subsidy



Textual Representation



Tasks

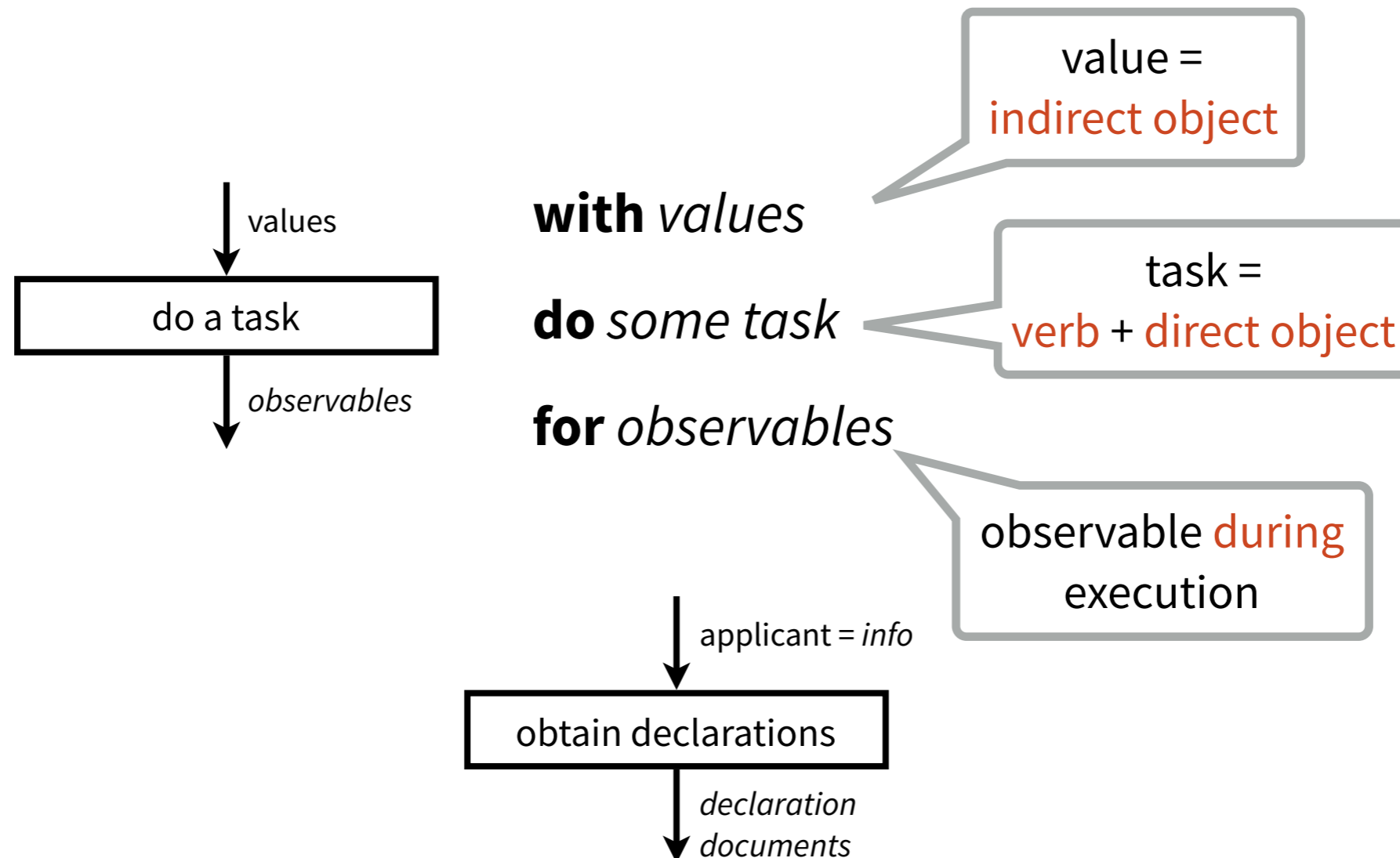


with a water hose **do** extinguish fire **for** success

with guns and canons **do** attack enemy **for** elimination

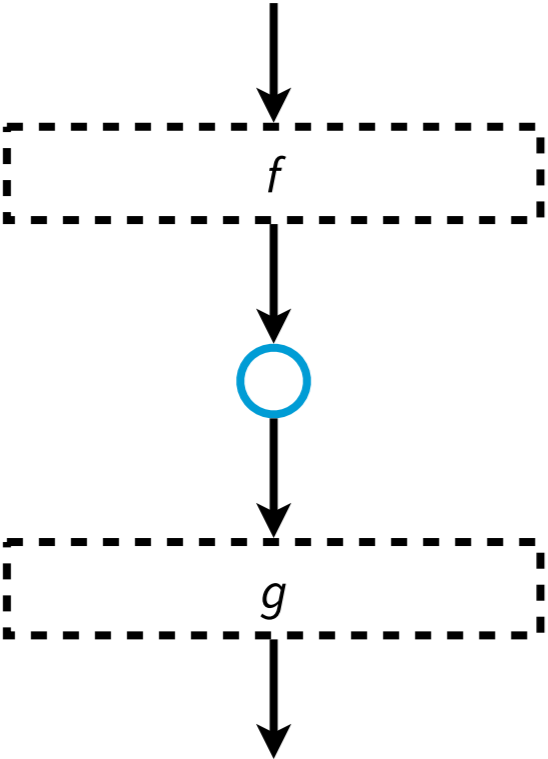
with applicant info **do** obtain declarations **for** declaration and documents

Tasks

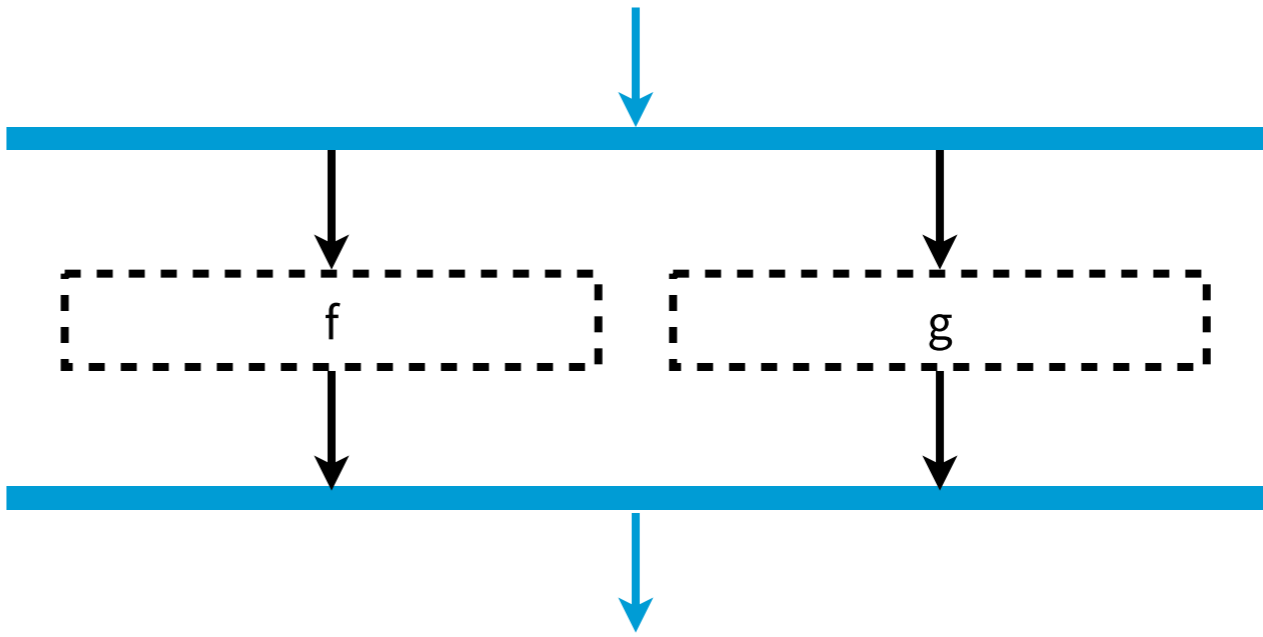


with applicant info **do** obtain declarations **for** declaration and documents

Connectives: sequence and parallel



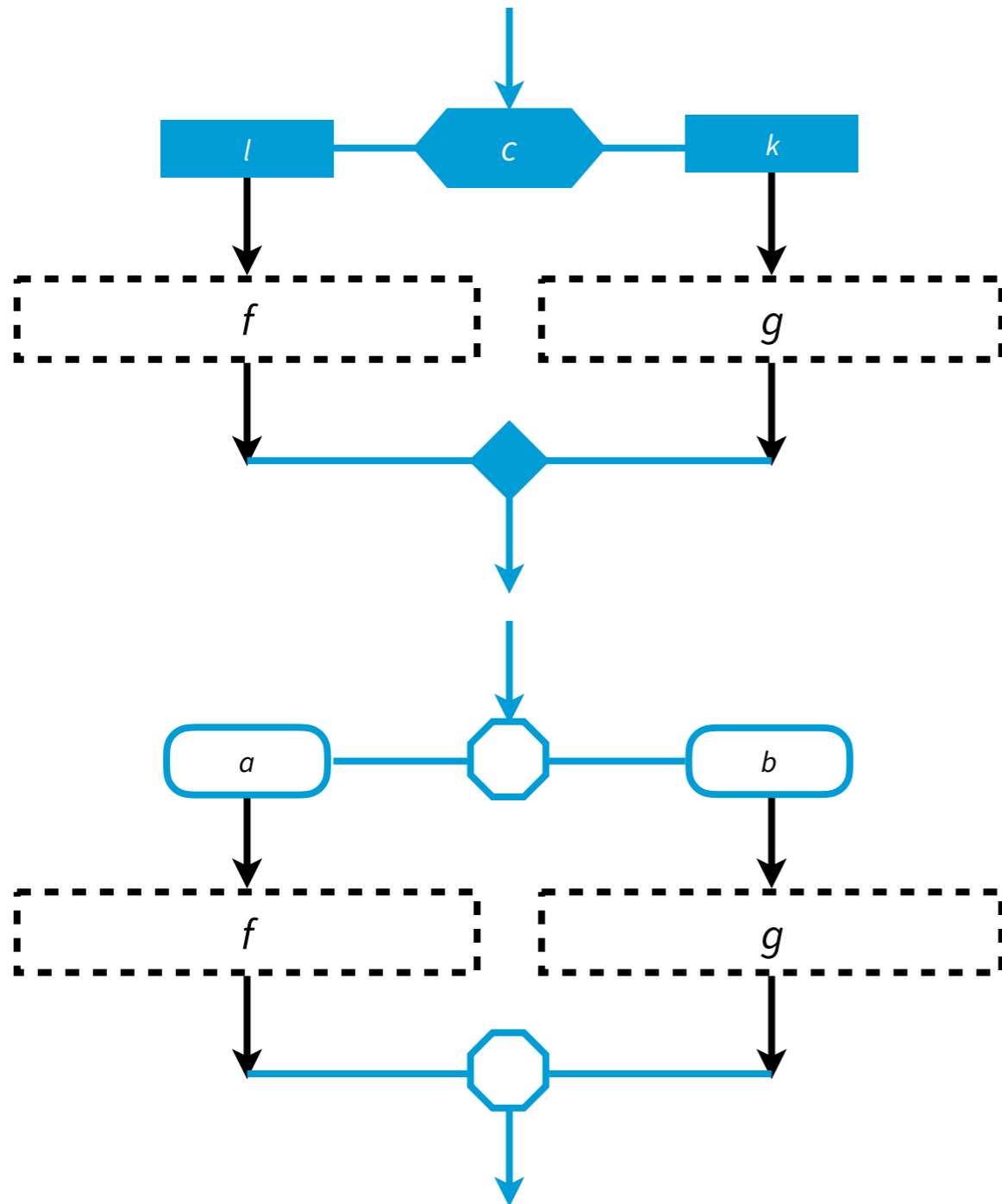
f then g



parallel
also
 f
:
also
 g



Connectives: choices



match c
case l **then**
 f
 $:$
case k **then**
 g

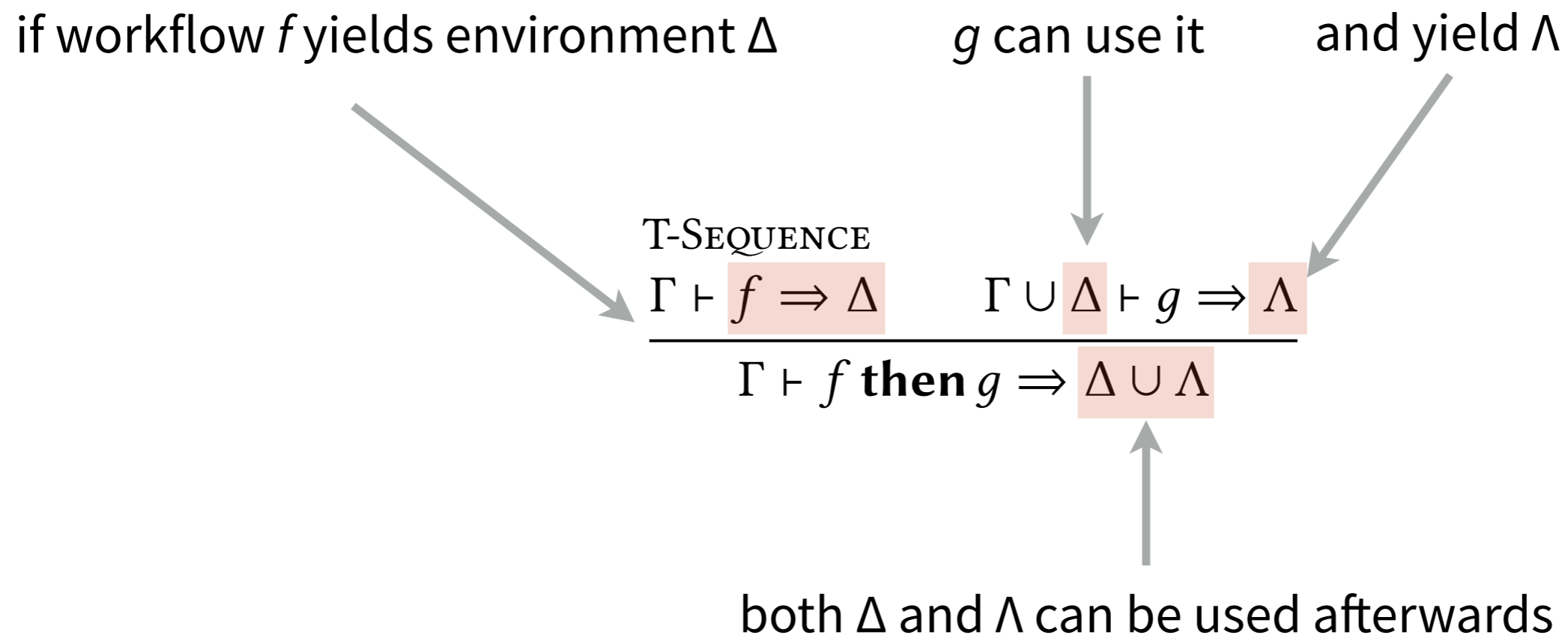
c are **conditions**
 matched to l and k
 by the system
 \Rightarrow **internal choice**

select
on a **then**
 f
 $:$
on b **then**
 g

a and b are **actions**
 selected by the user
 \Rightarrow **external choice**

Type checking: connectives

- Connectives make sure all resources are in **scope**



Type checking: tasks

- Based on **records** with labeled values

if all values v_l are of type τ_l and o is an operation taking τ_l 's to σ_k 's

T-APPLY

$$\frac{\Gamma \vdash o : \{\overline{l} : \tau_l\} \rightarrow \{\overline{k} : \sigma_k\} \quad \Gamma \vdash v : \{\overline{l} : \tau_l\} \quad \vdash p : \{\overline{k} : \sigma_k\} \Rightarrow \Delta}{\Gamma \vdash \mathbf{with } v \mathbf{ do } o \mathbf{ for } p \Rightarrow \Delta}$$

pattern p binds values v_k of type σ_k in a new environment Δ



Compiling

- **Clean** as host language
- using **monadic** operations
- and **built-in** *iTask* operators

$$\{\text{with } v \text{ do } o \text{ for } p\} = \{o\} \{v\} \gg= \setminus \{p\} \rightarrow$$
$$\{\text{parallel } \overline{\text{also } f_i}\} = \text{allTasks} \begin{array}{l} [\text{ /* for each } i \text{ */} \\ \{f_i\} \\ \text{return } (\text{Bri } \{Bound(f_i)\}) \\] \gg= \setminus \\ [\text{ /* for each } i \text{ */} \\ \text{Bri } \{Bound(f_i)\}] \rightarrow \end{array}$$


Wrap up



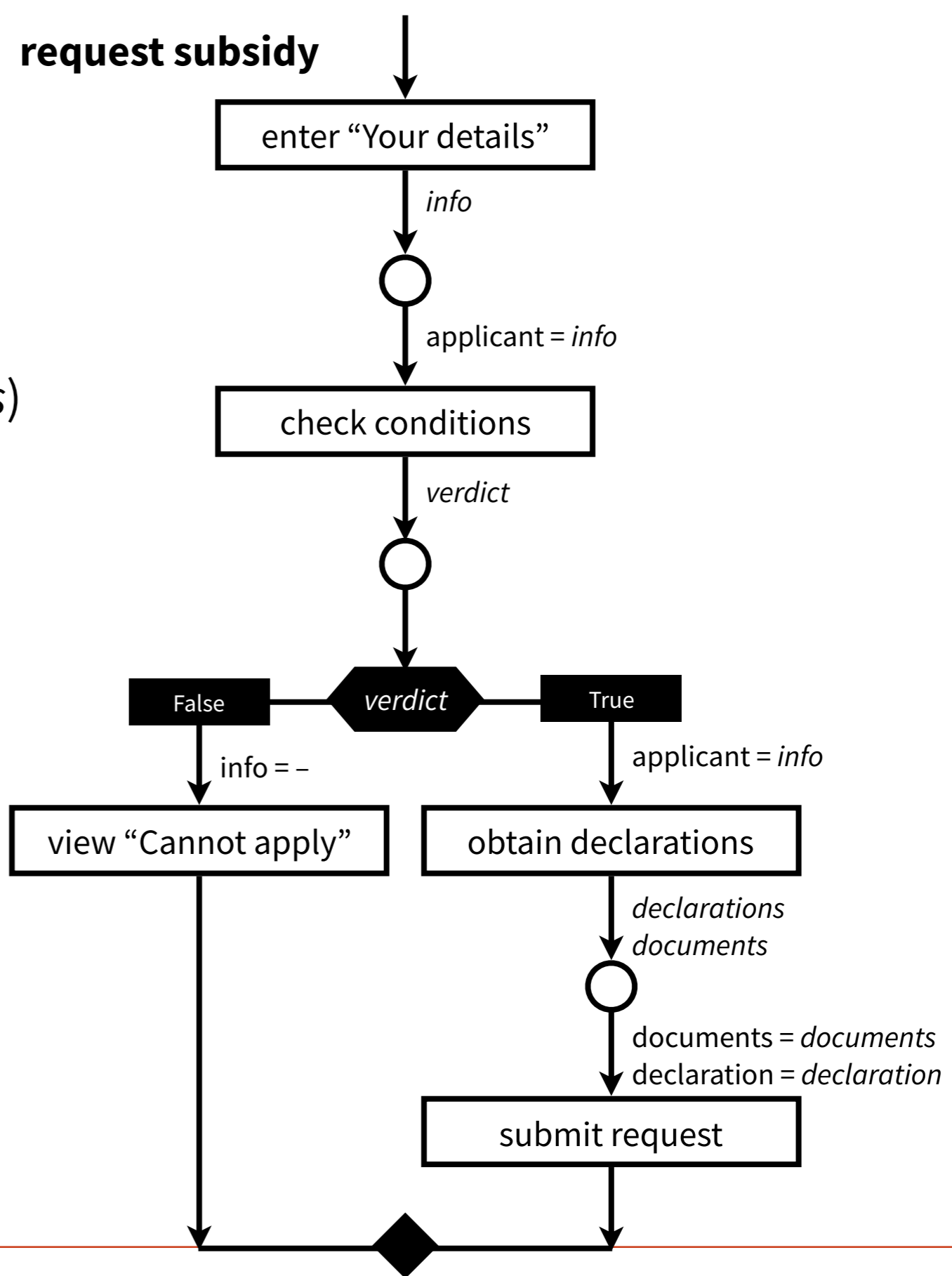
Conclusion

What did we see?

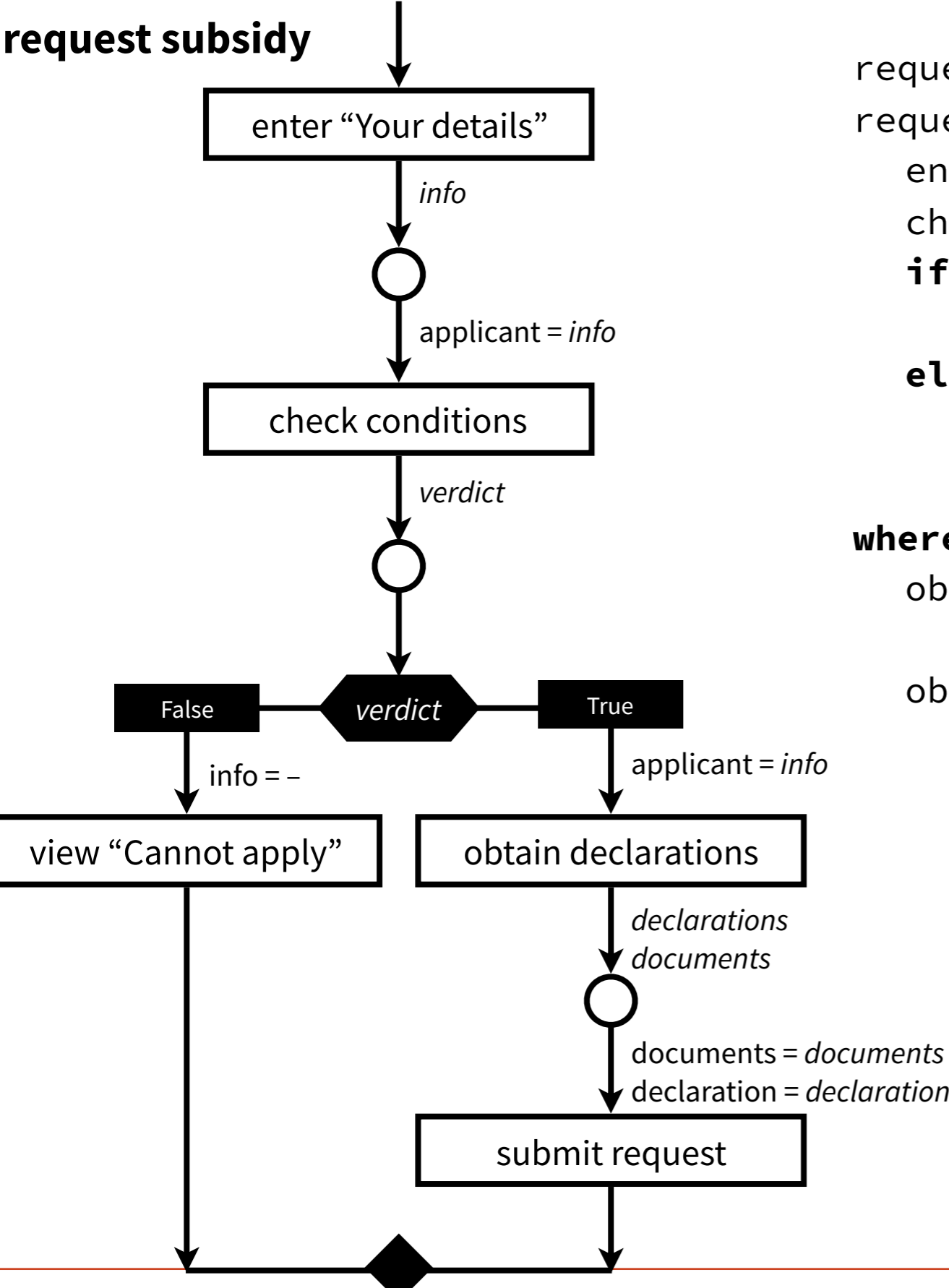
- One language (DSL, subset of *iTasks*)
- Two representations:
 - Visual
 - Textual

How did we use it?

- Visual design tool
- Model tasks and workflows
- Modular and assistive
- Supported by a type system
- Compilable to *iTasks*



request subsidy



```

request_subsidy :: () -> Task ()
request_subsidy () =
  enter "Your details" >>= \info ->
  check_conditions info >>= \verdict ->
  if not verdict then
    view "Cannot apply" ()
  else
    obtain_declarations info >>= \decl, docs ->
    submit_request decl docs
  where
    obtain_declarations :: CitizenInfo
      -> Task (CompensationDocs, ContractorDecl)
    obtain_declarations applicant =
      (provide_documents)
      -&&-
      (select_company >>= \company ->
        provide_declaration info contractor)
  
```