Filesystems deserve verification too!

Gabriele Keller, Toby Murray, Sidney Amani,
Liam O'Connor, Zilin Chen, Leonid Ryzhyk
Gerwin Klein, Gernot Heiser

Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

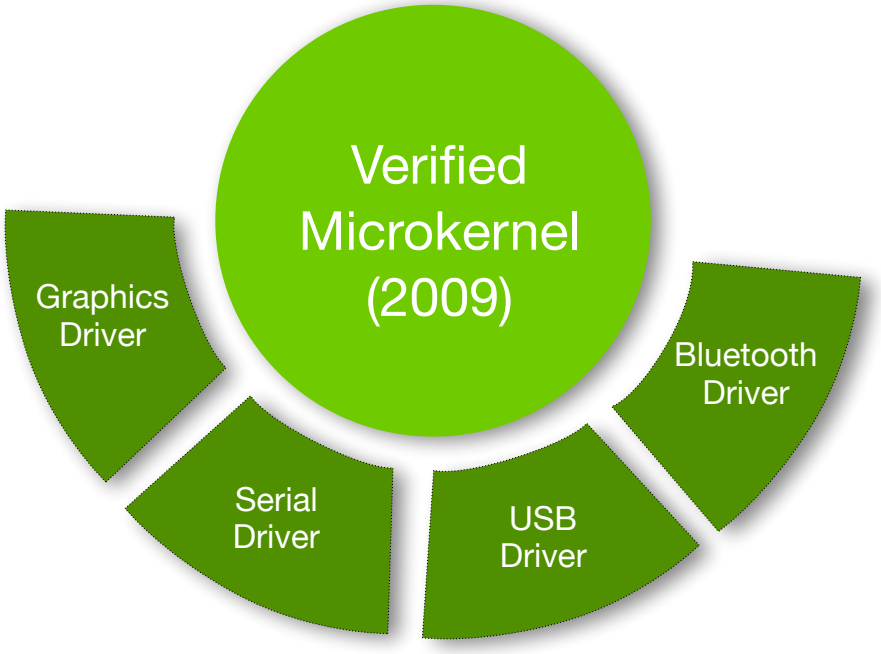
NICTA Funding and Supporting Members and Partners
Australian National University
UNSW
NSW Trade & Investment
State Government Victoria
The University of Melbourne
The University of Sydney
Queensland Government
Griffith University
QUT
The University Of Queensland

Towards a trustworthy system

Towards a trustworthy system

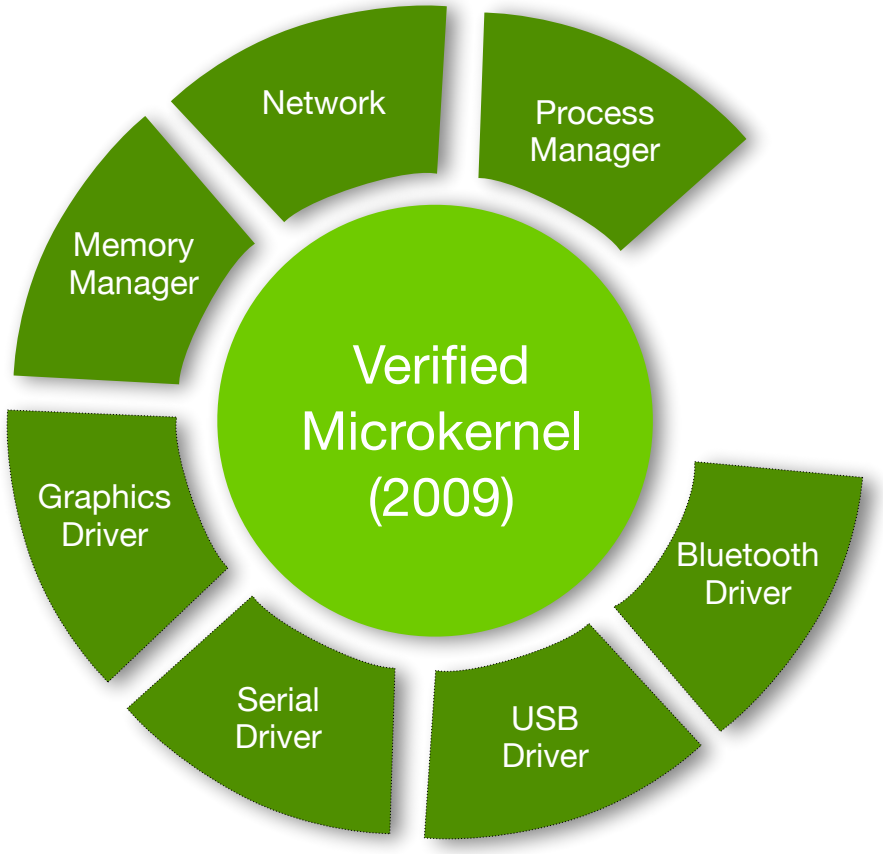
Towards a trustworthy system

NICTA



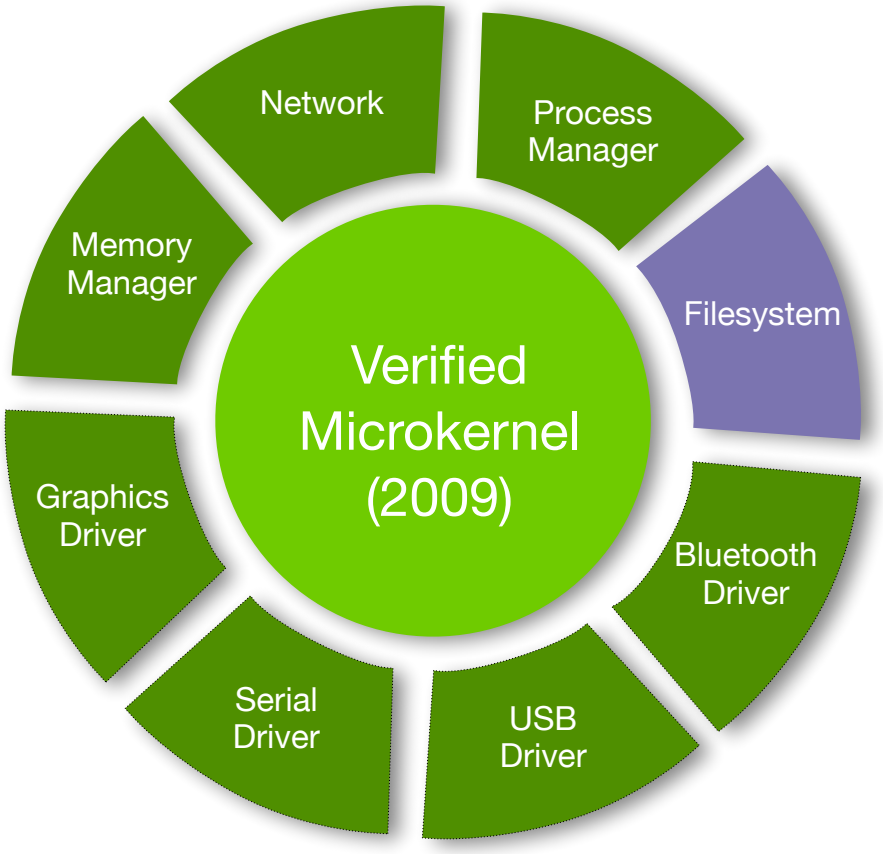
Towards a trustworthy system

NICTA



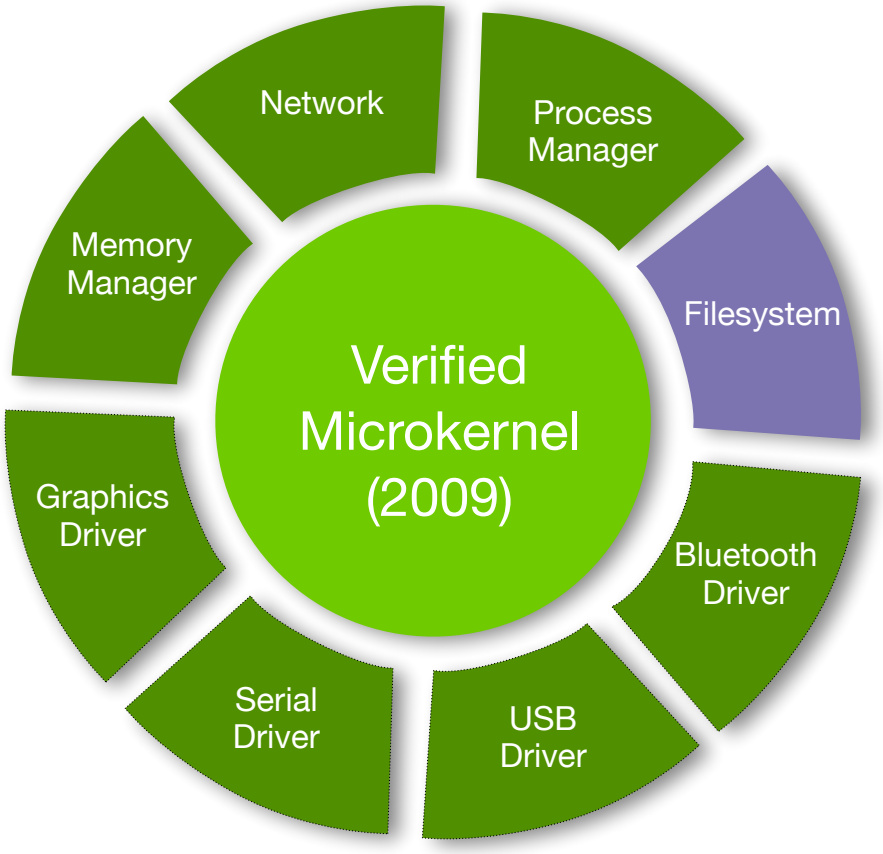
Towards a trustworthy system

NICTA



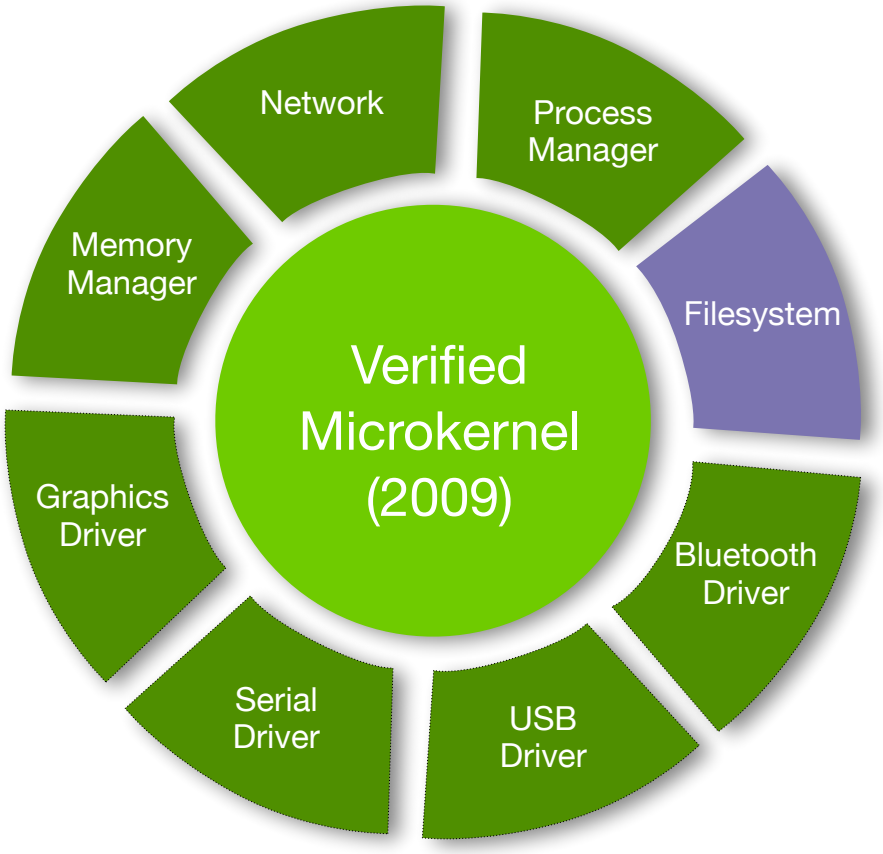
Towards a trustworthy system

NICTA



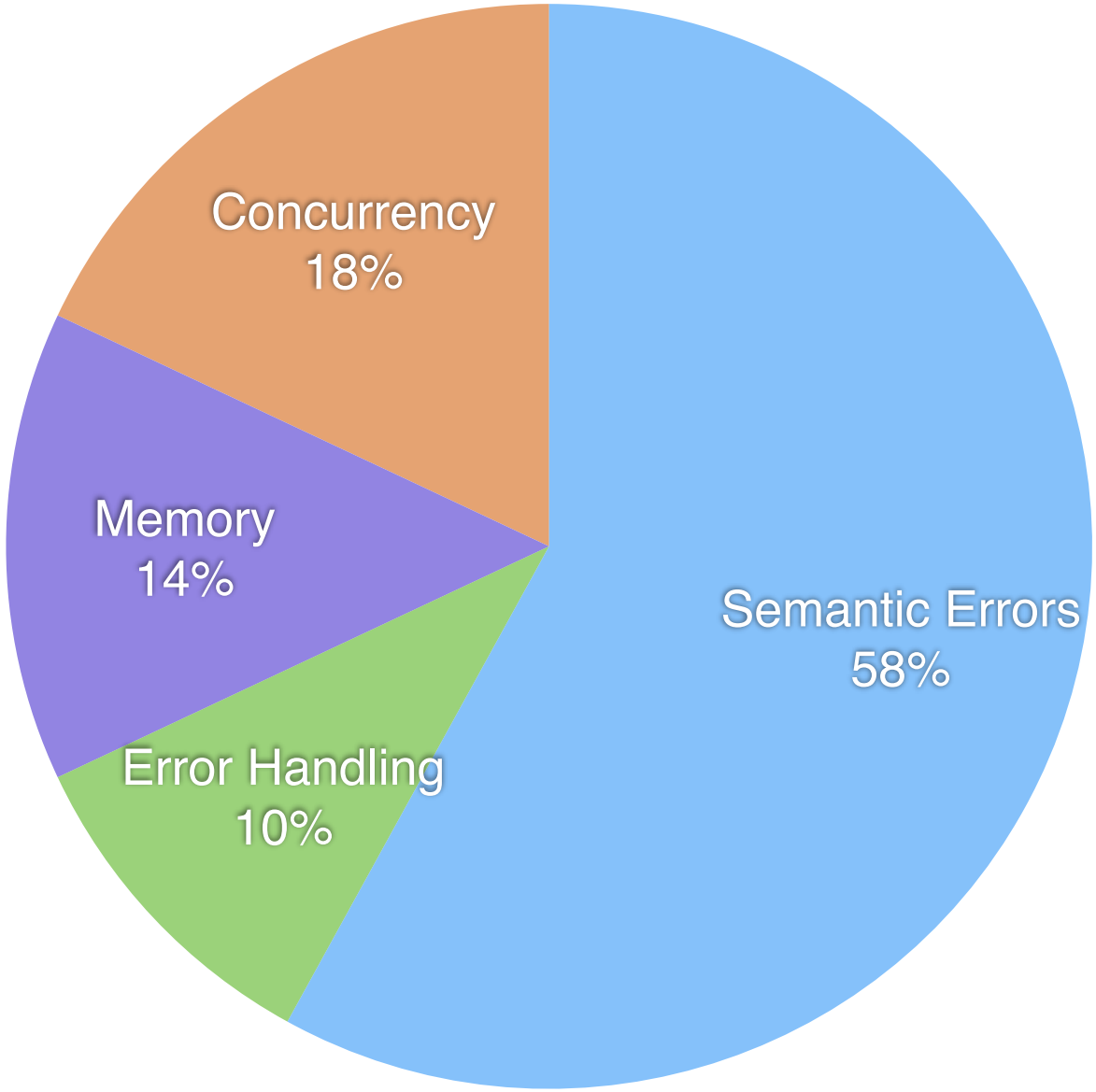
Towards a trustworthy system

NICTA

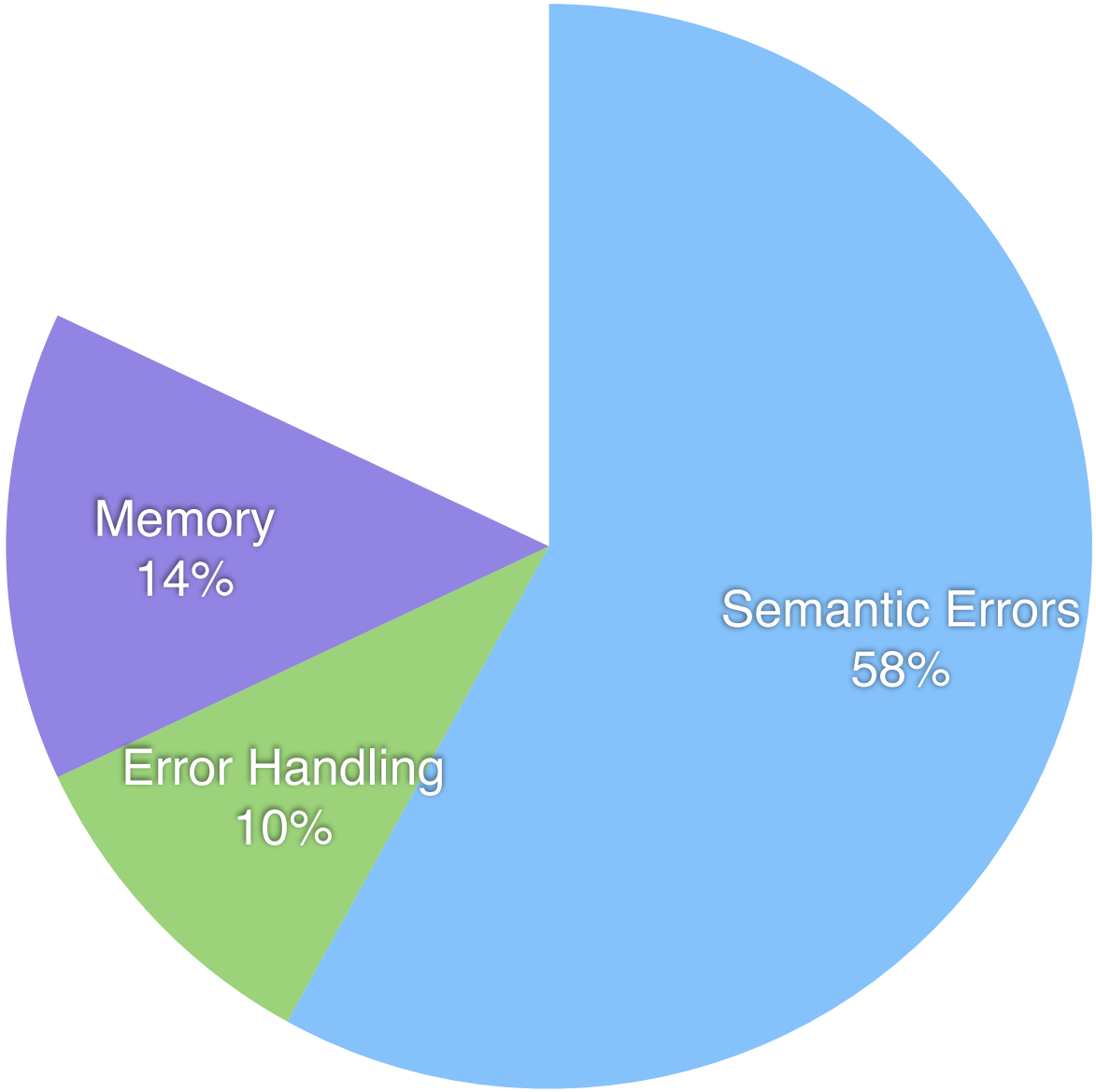


File System Bugs [Lu et al]

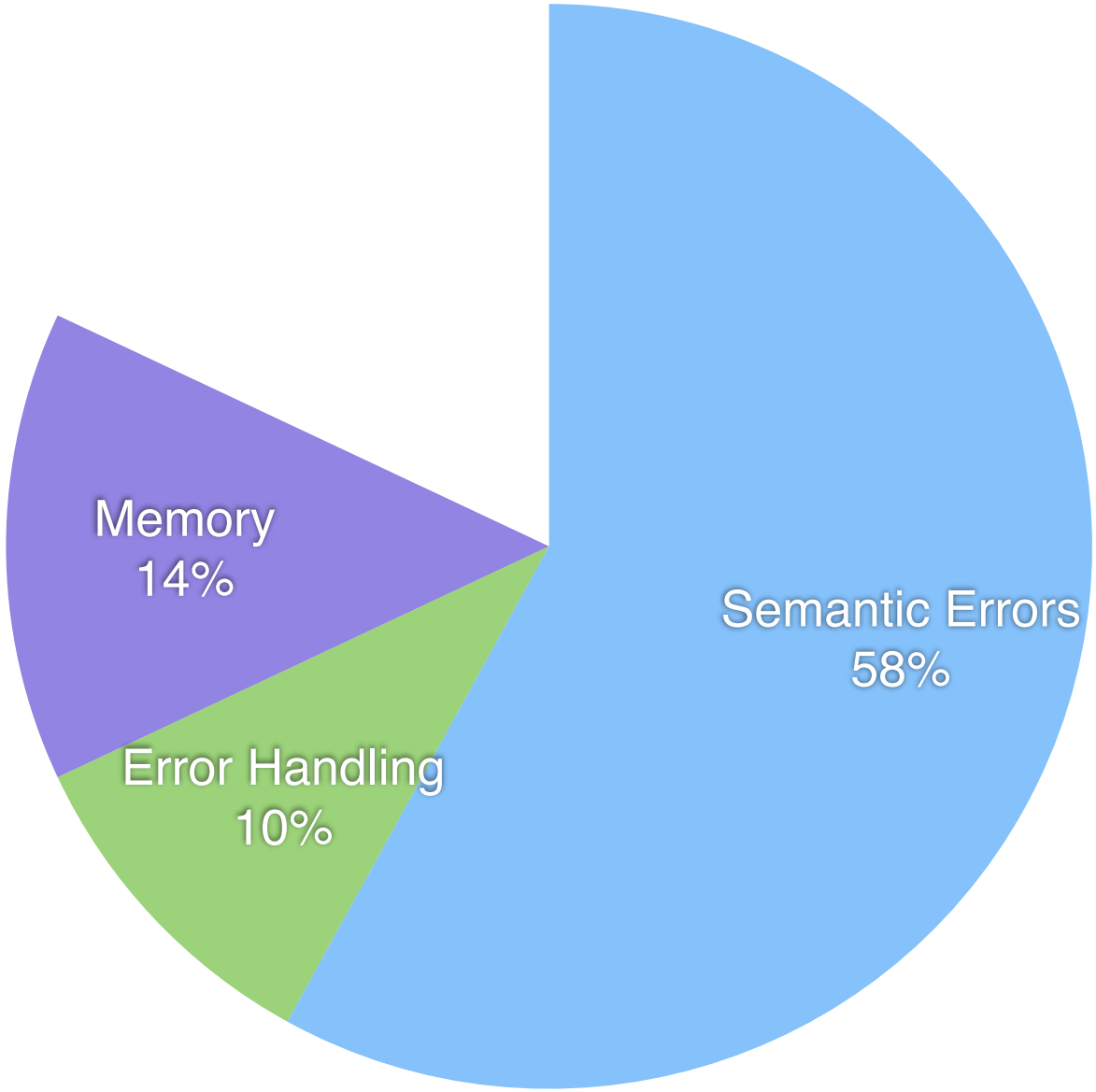
NICTA



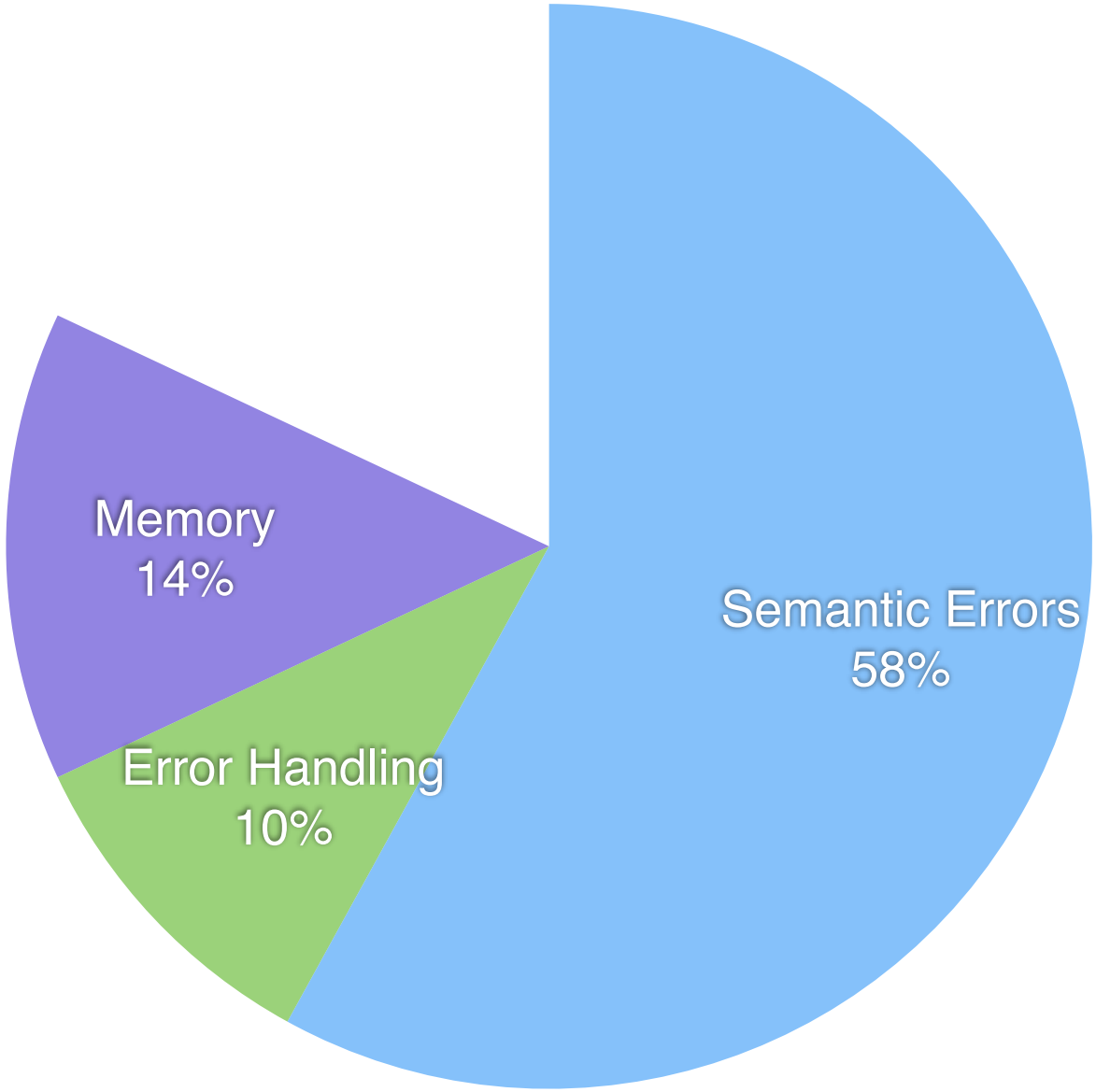
File System Bugs [Lu et al]



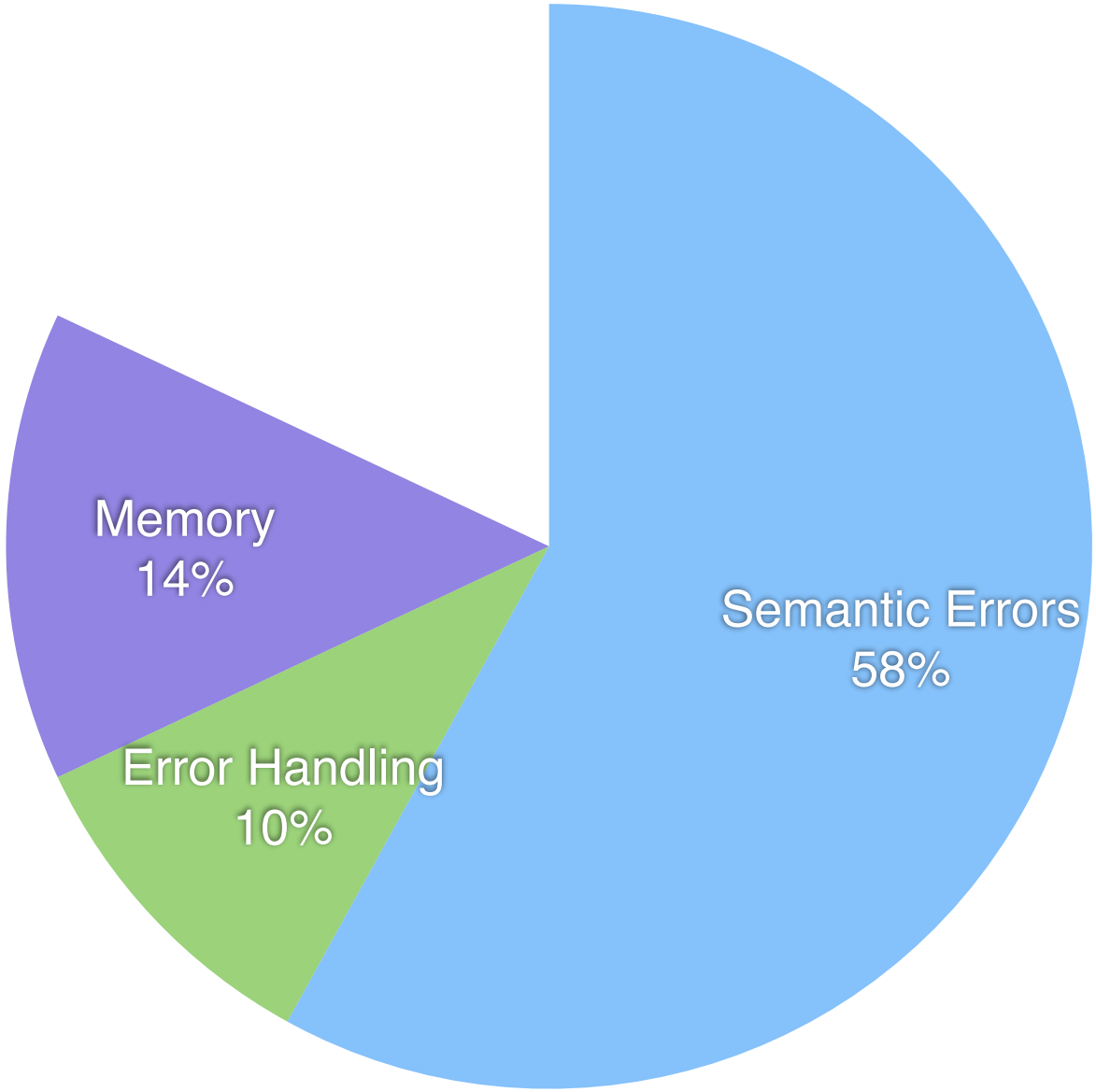
File System Bugs [Lu et al]



File System Bugs [Lu et al]



File System Bugs [Lu et al]



NICTA

Functional Specification

Filesystem Code

NICTA

Functional Specification

Refinement Proof

Filesystem Code

NICTA

Functional Specification

Refinement Proof

Low-level Specification

Refinement Proof

Filesystem Code

NICTA

Functional Specification

Refinement Proof

relatively easy to do manually
hard to automate

Low-level Specification

Refinement Proof

Filesystem Code

NICTA

Functional Specification

Refinement Proof

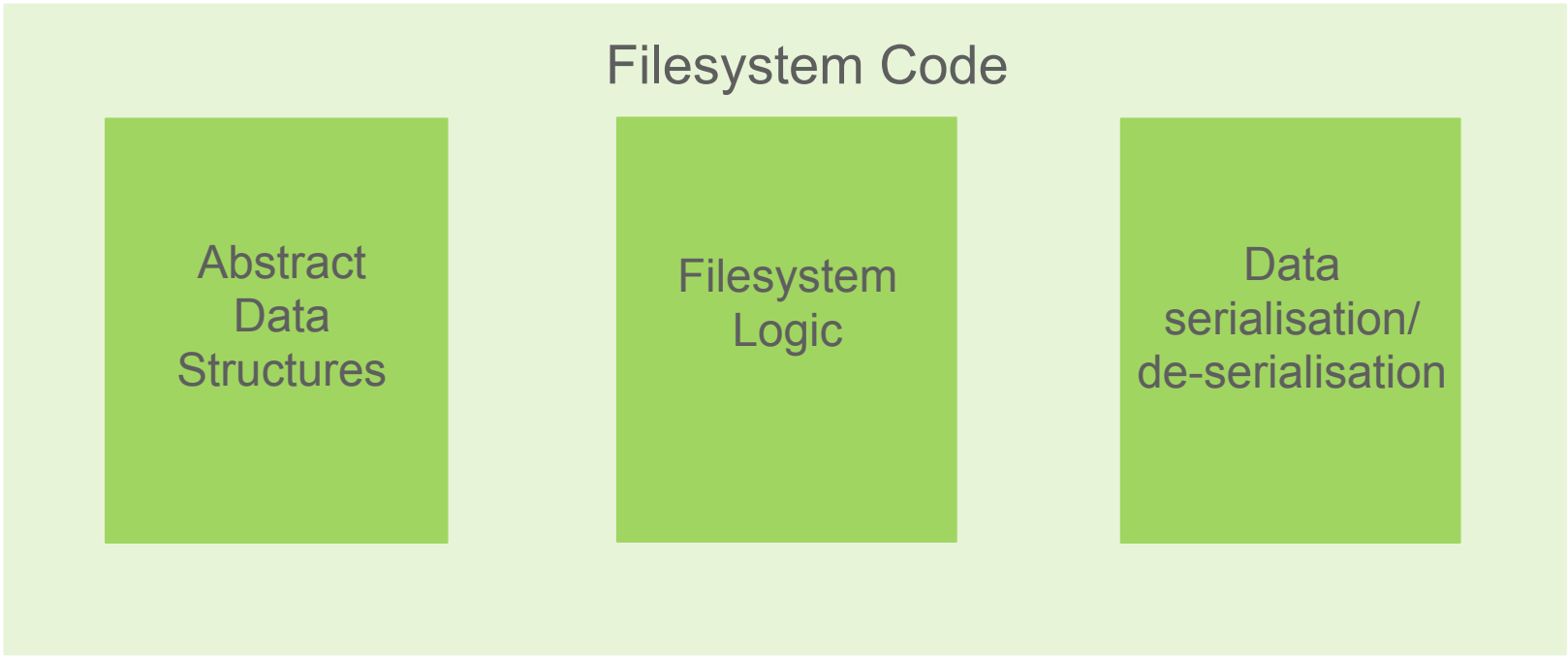
relatively easy to do manually
hard to automate

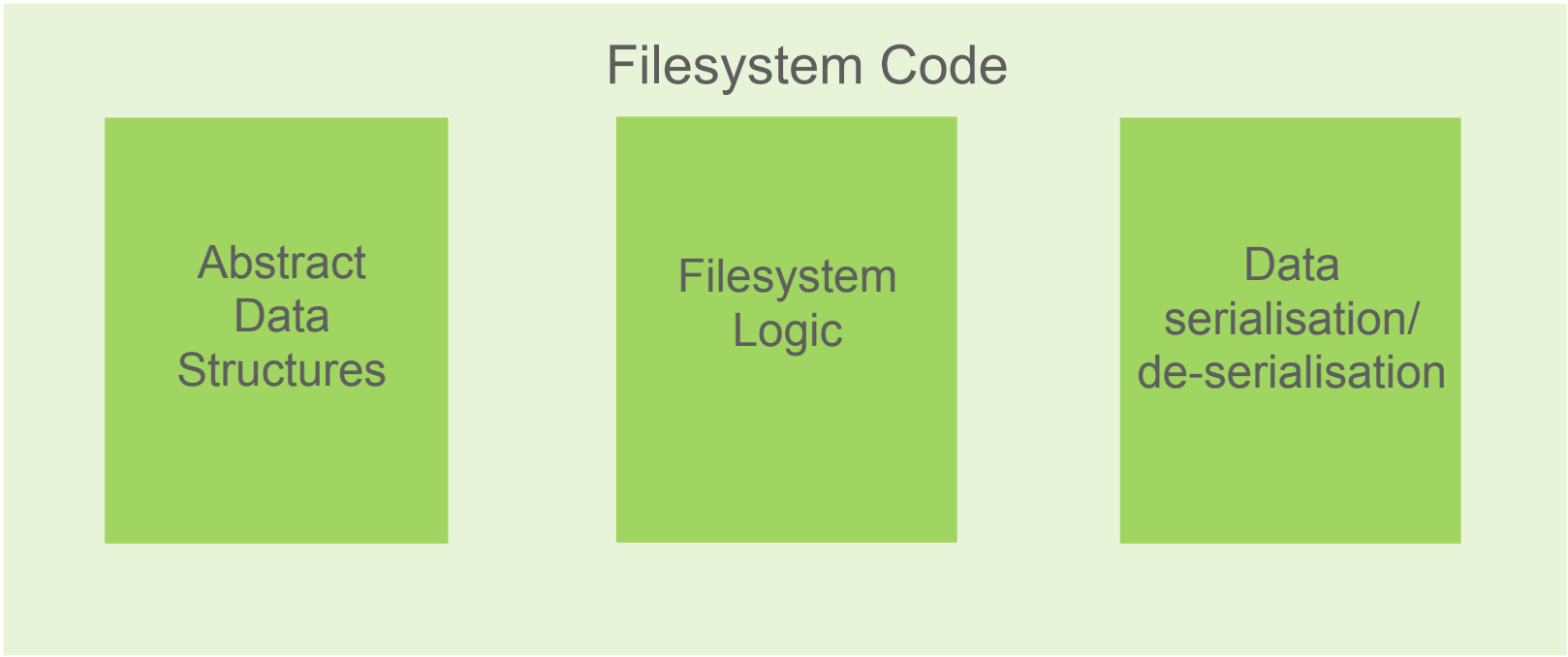
Low-level Specification

Refinement Proof

good candidate for automation

Filesystem Code





Isabelle Functional Specification

Filesystem

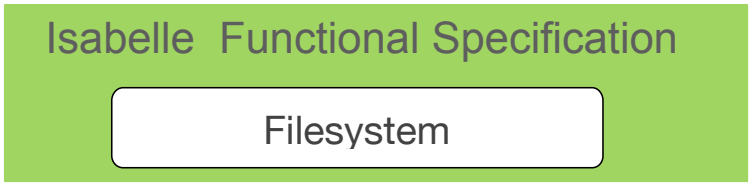
Isabelle Functional Specification

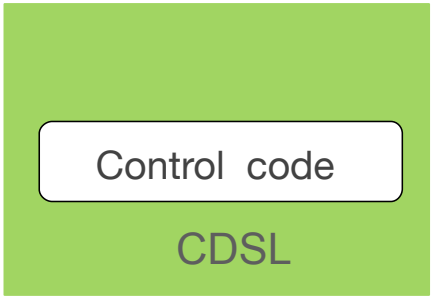
Filesystem

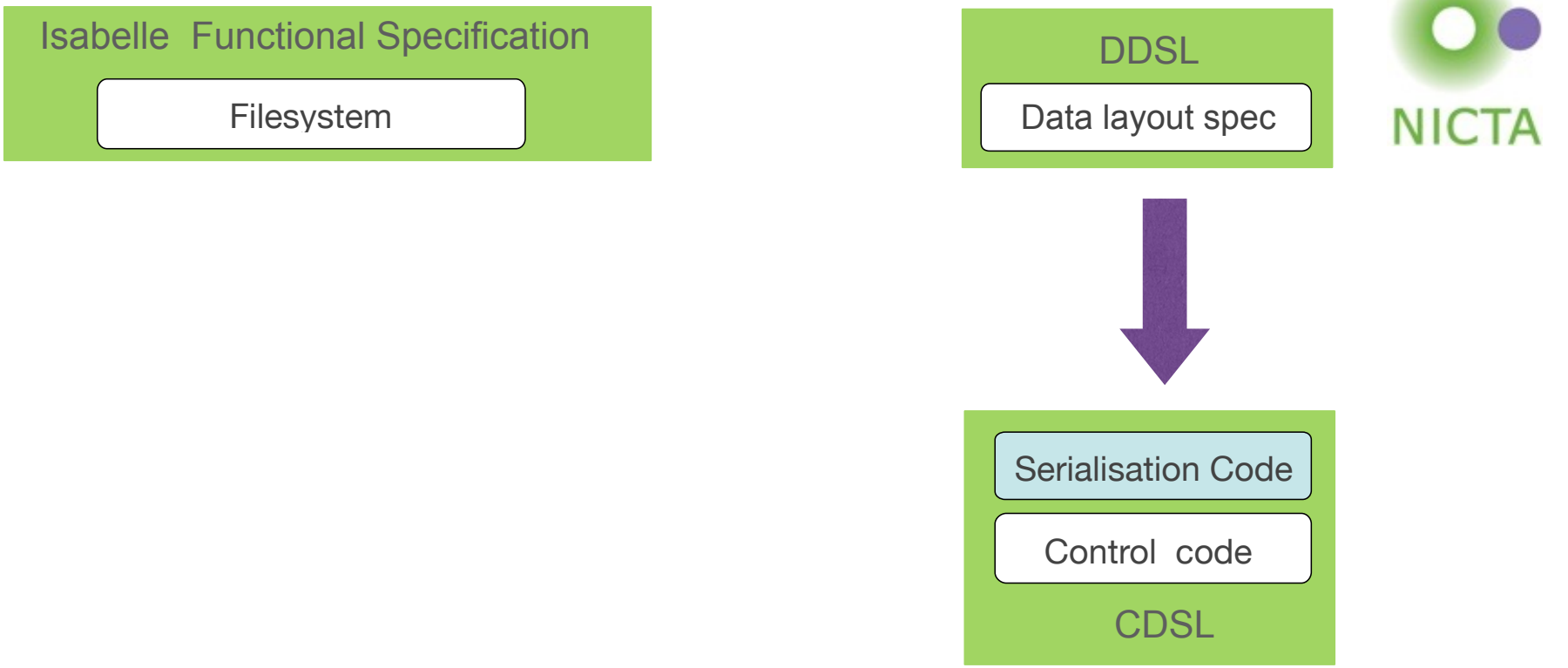
DDSL

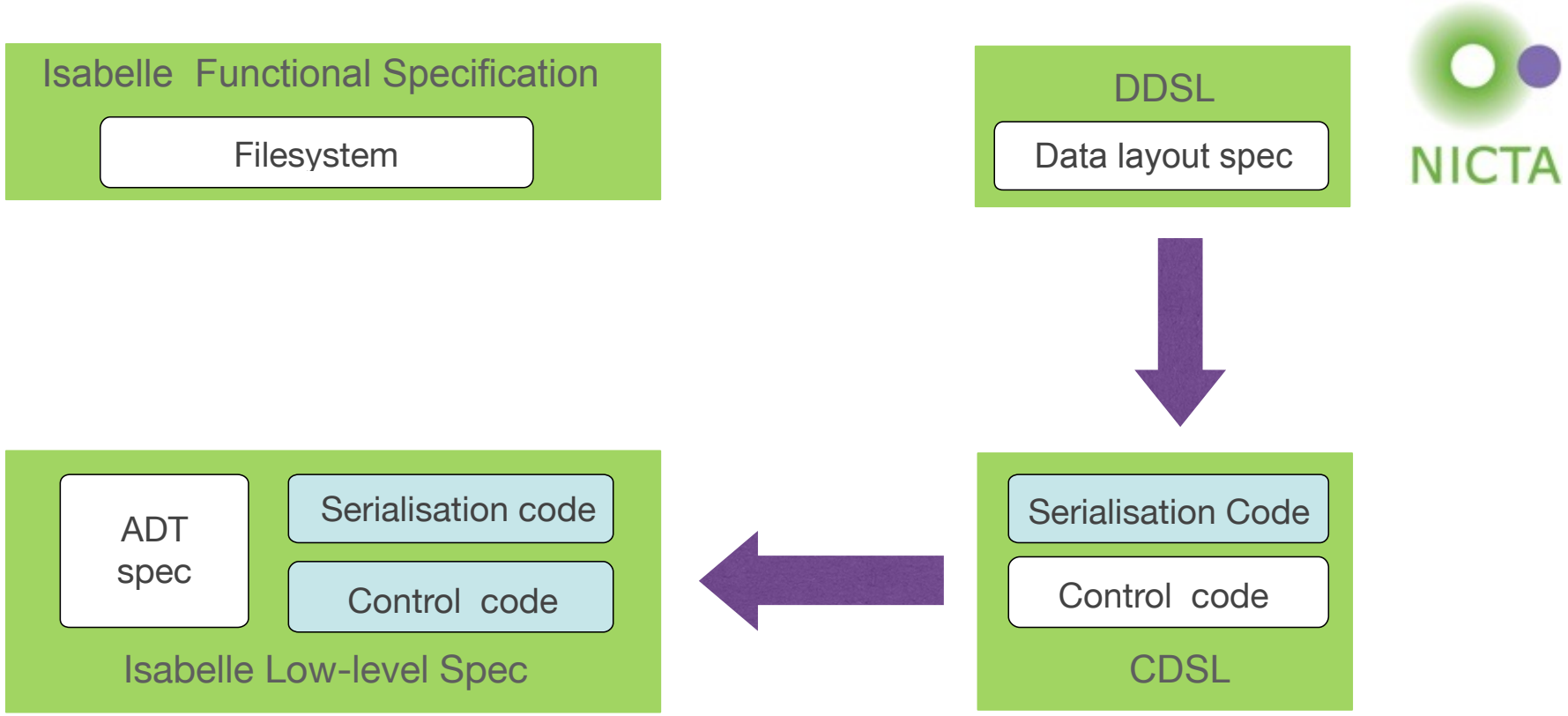
Data layout spec

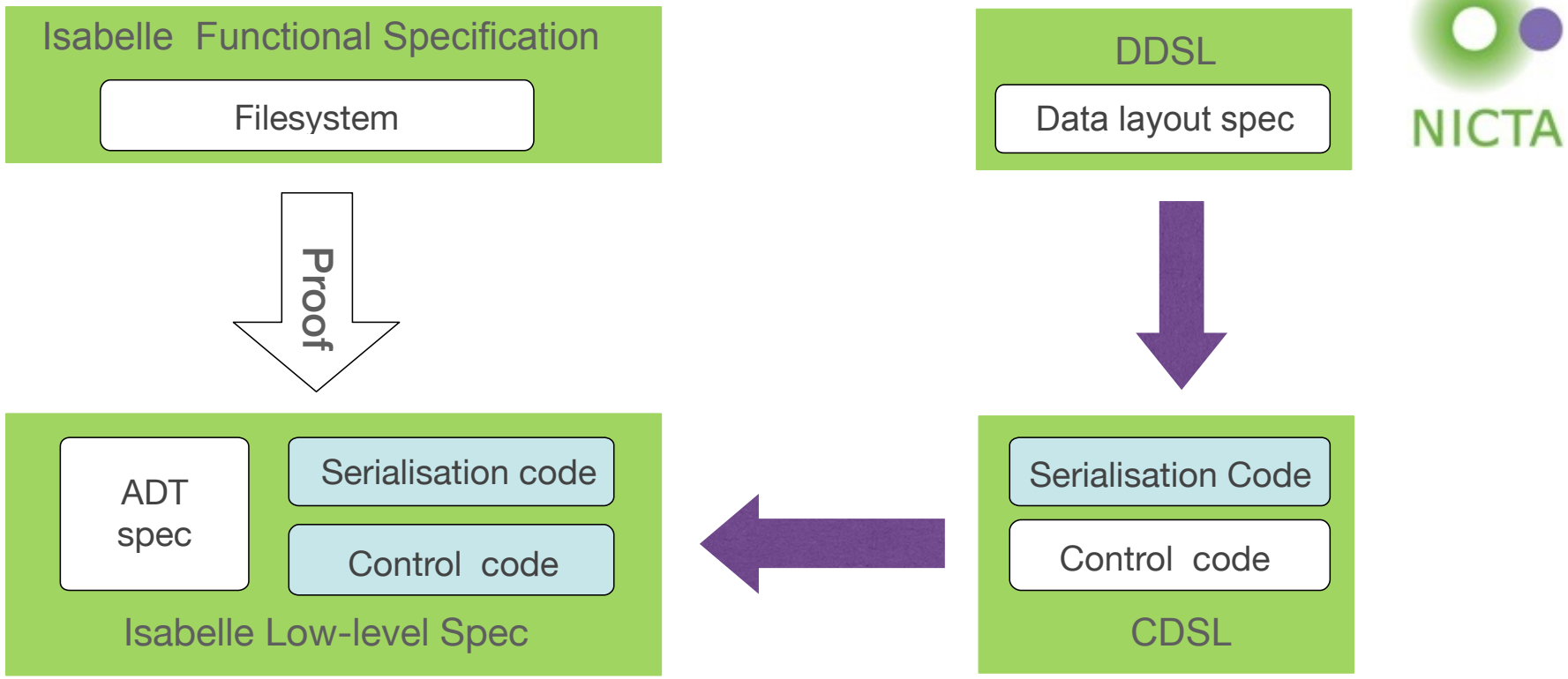
NICTA

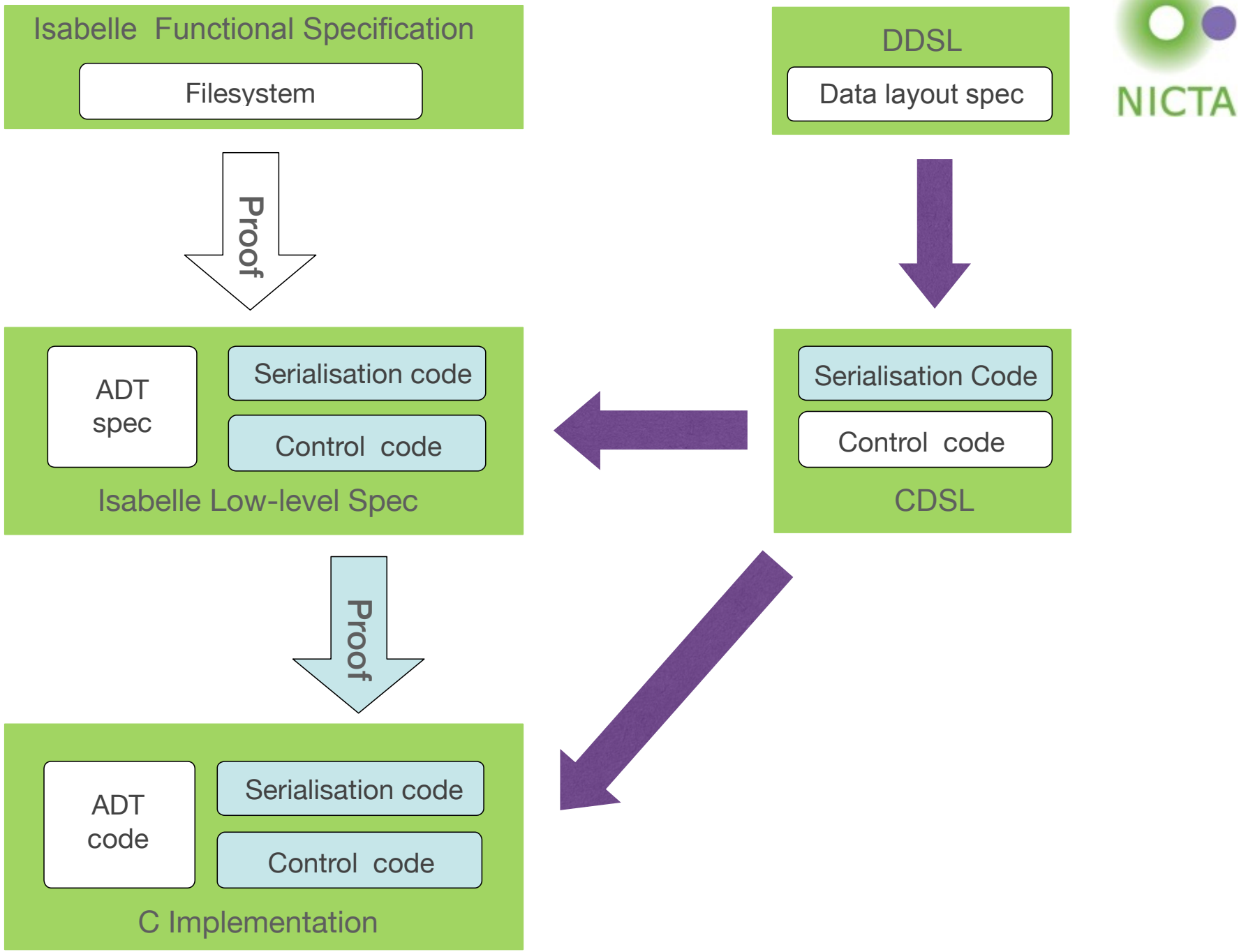


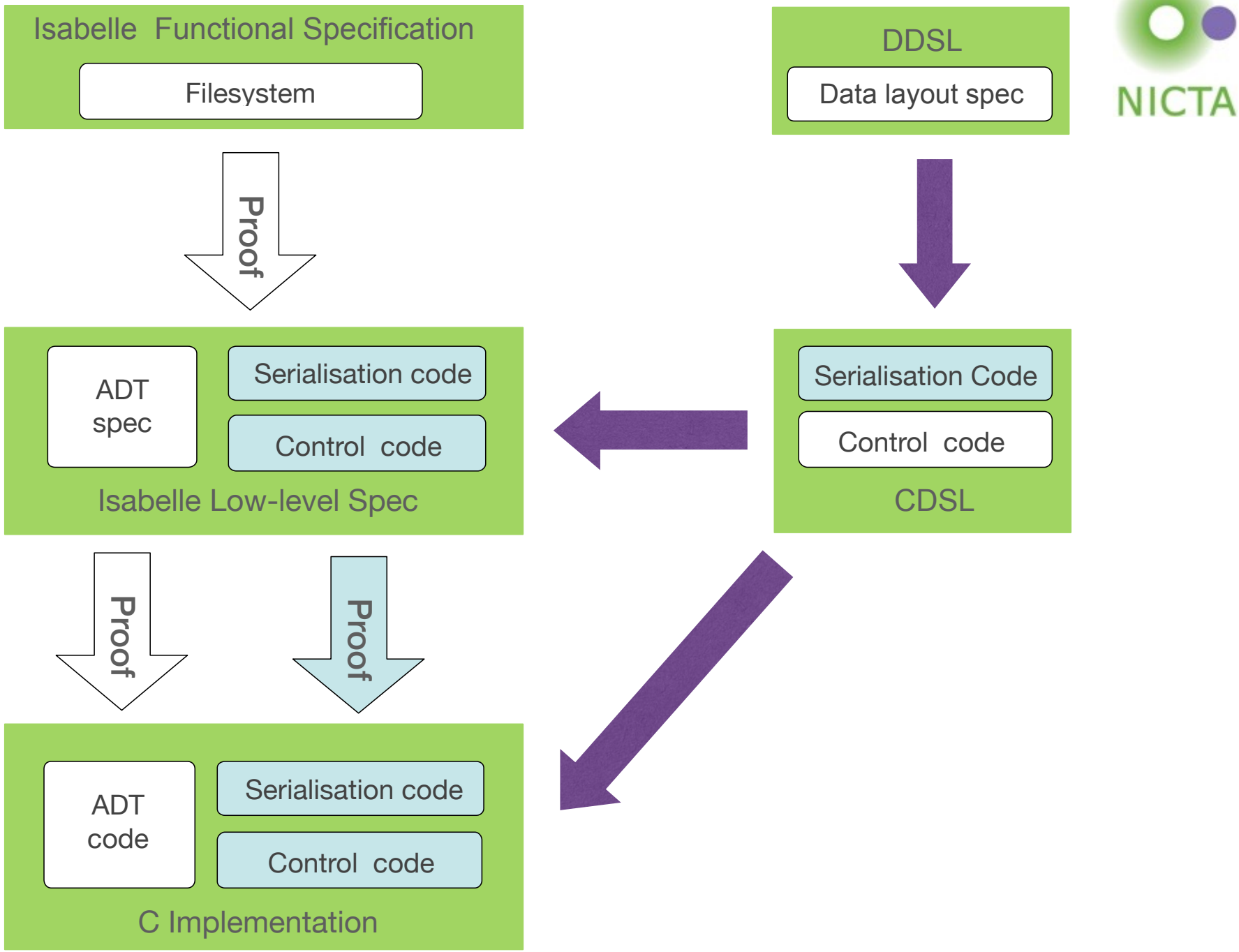



Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
Serialisation Code
Control code
CDSL


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL

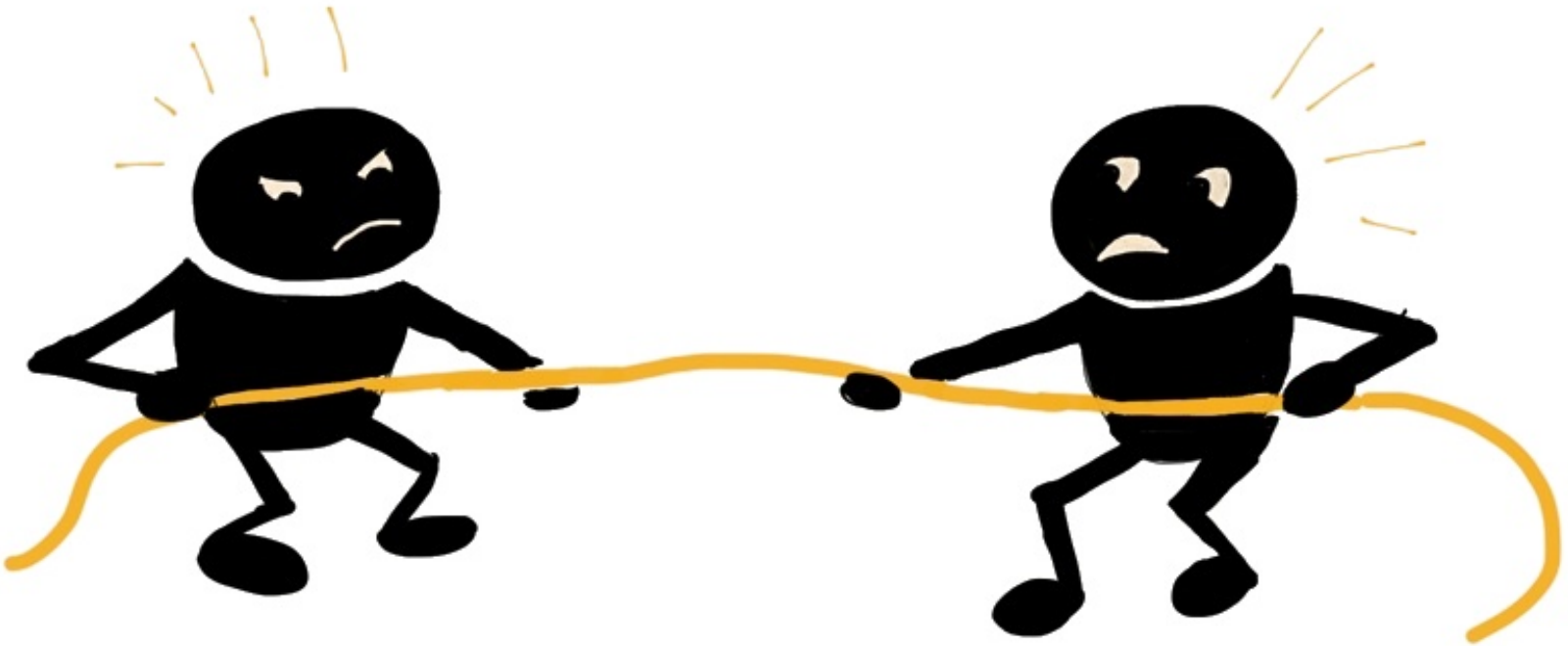

Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
ADT code
Serialisation code
Control code
C Implementation


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation

Design of the Control DSL

NICTA

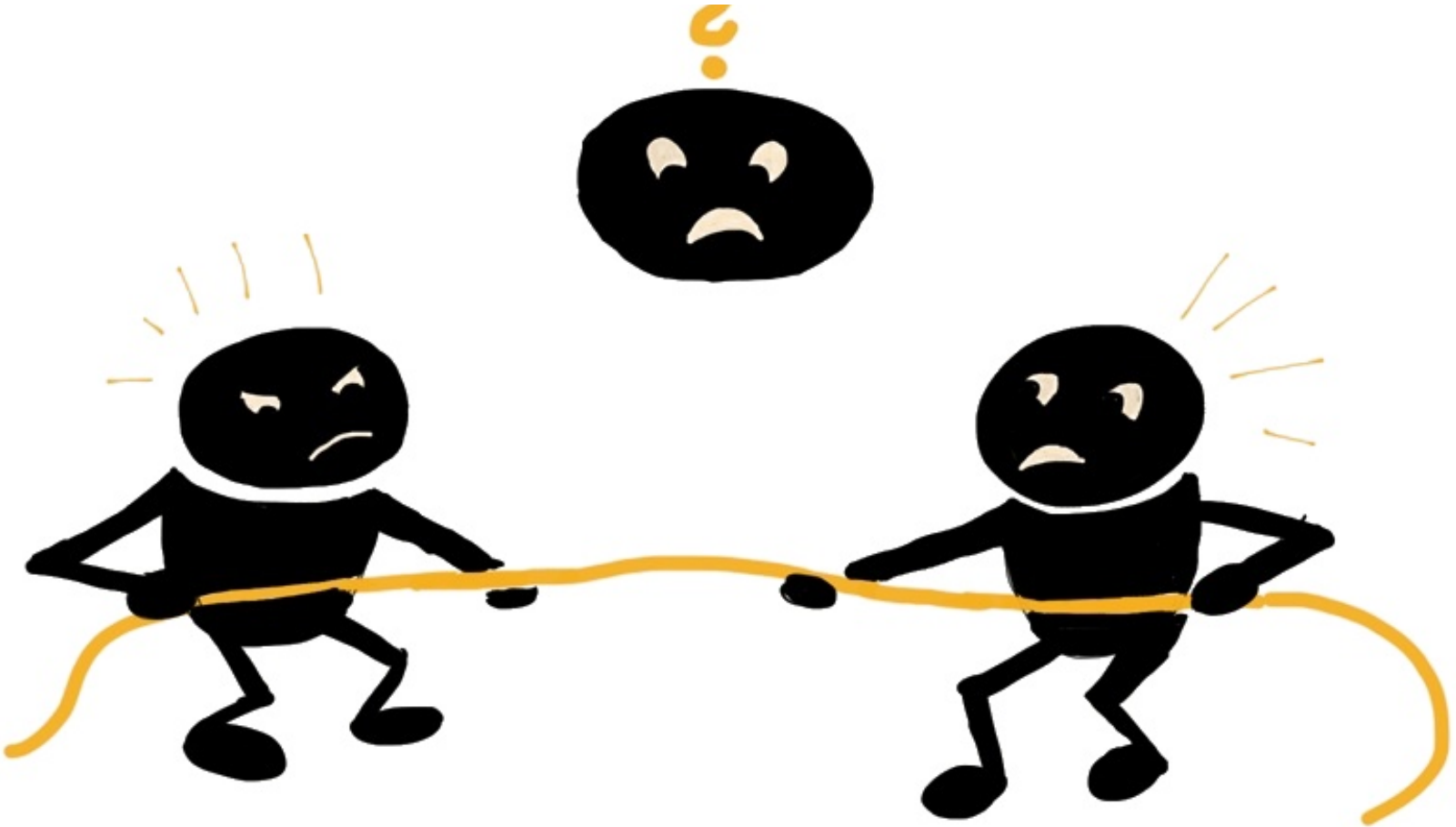


Systems:
- efficiency
- destructive updates
- expressiveness
- concise

Verification:
- controlled side effects
- memory & type safety
- termination
- equational reasoning

Design of the Control DSL

NICTA



Systems:
- efficiency
- destructive updates
- expressiveness
- concise

Verification:
- controlled side effects
- memory & type safety
- termination
- equational reasoning

Updates

NICTA

```
f(x) = let
  x1 = update1 x
  x2 = update2 x1
in x2
```

Updates

NICTA

```
f(x) = let
   x1 = update1 x
   x2 = update2 x1
in x2
```

x no longer used

Updates

NICTA

```
f(x) = let
   x1 = update1 x
   x2 = update2 x1
in x2
```

x no longer used

Updates

NICTA

```
f(x) = let
  x1 = update!1 x
  x2 = update2 x1
in x2
```

Updates

NICTA

```
f(x) = let
  x1 = update!1 x
  x2 = update2 x1
in x2
```

x_1 no longer used

Updates

NICTA

```
f(x) = let
  x1 = update!1 x
  x2 = update2 x1
in x2
```

x_1 no longer used

Updates

NICTA

```
f(x) = let
  x1 = update!1 x
  x2 = update!2 x1
in x2
```

Memory Management

NICTA

```
f(x) = let
   (x', xc) = copy x
   (ok, x") = foo x'
in if ok
   then x"
   else xc
```

Memory Management

NICTA

```
f(x) = let
   (x', xc) = copy x
   (ok, x") = foo x'
in if ok
   then x"
   else xc
```

Error: $\mathbf{x_c}$ not used

Error: **x"** not used

Memory Management

NICTA

```
f(x) = let
   (x', xc) = copy x
   (ok, x") = foo x'
in if ok
   then x"
   else xc
```

Error: $\mathbf{x_c}$ not used

Error: **x"** not used

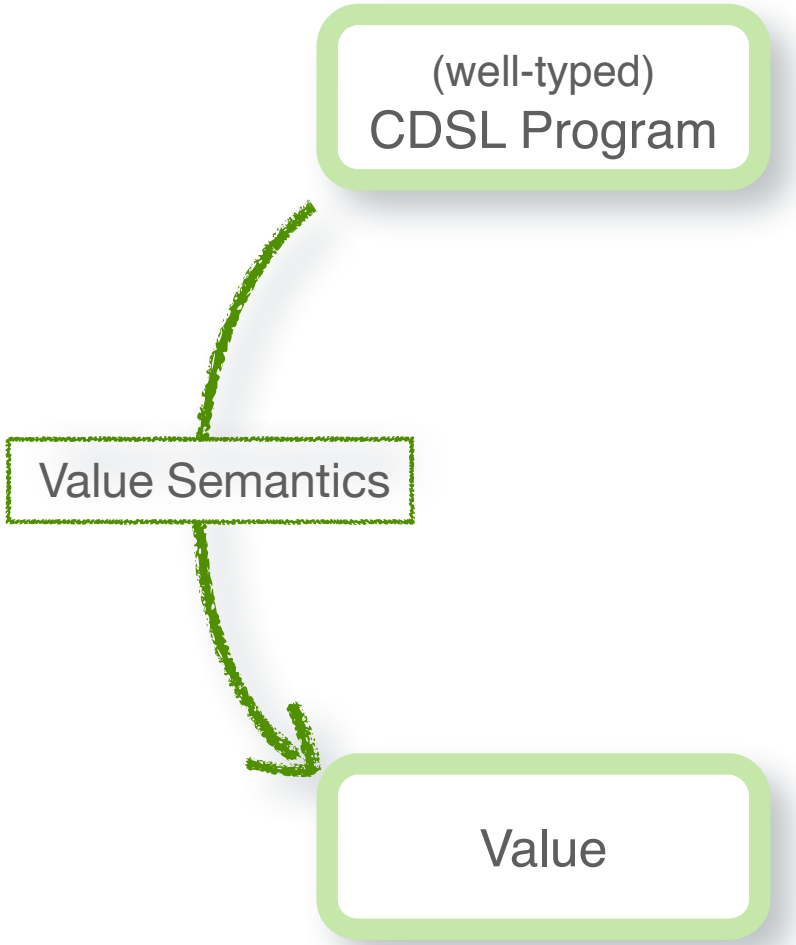
Memory Management

NICTA

```
f(x) = let
   (x', xc) = copy x
   (ok, x") = foo x'
in if ok
    then free xc ; x"
    else free x" ; xc
```

Value and Update Semantics of CDSL

Value and Update Semantics of CDSL

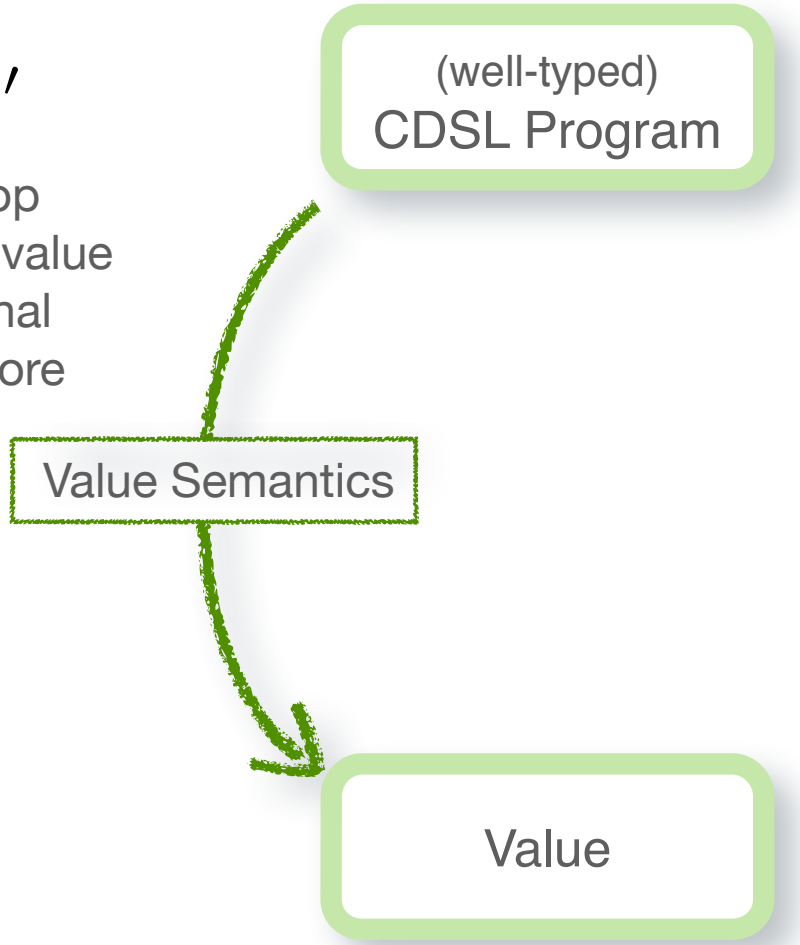


Value and Update Semantics of CDSL

NICTA

$$\Gamma \vdash e \mapsto e'$$

- `free` is a no-op
- Everything by value
- Purely functional
- No mutable store

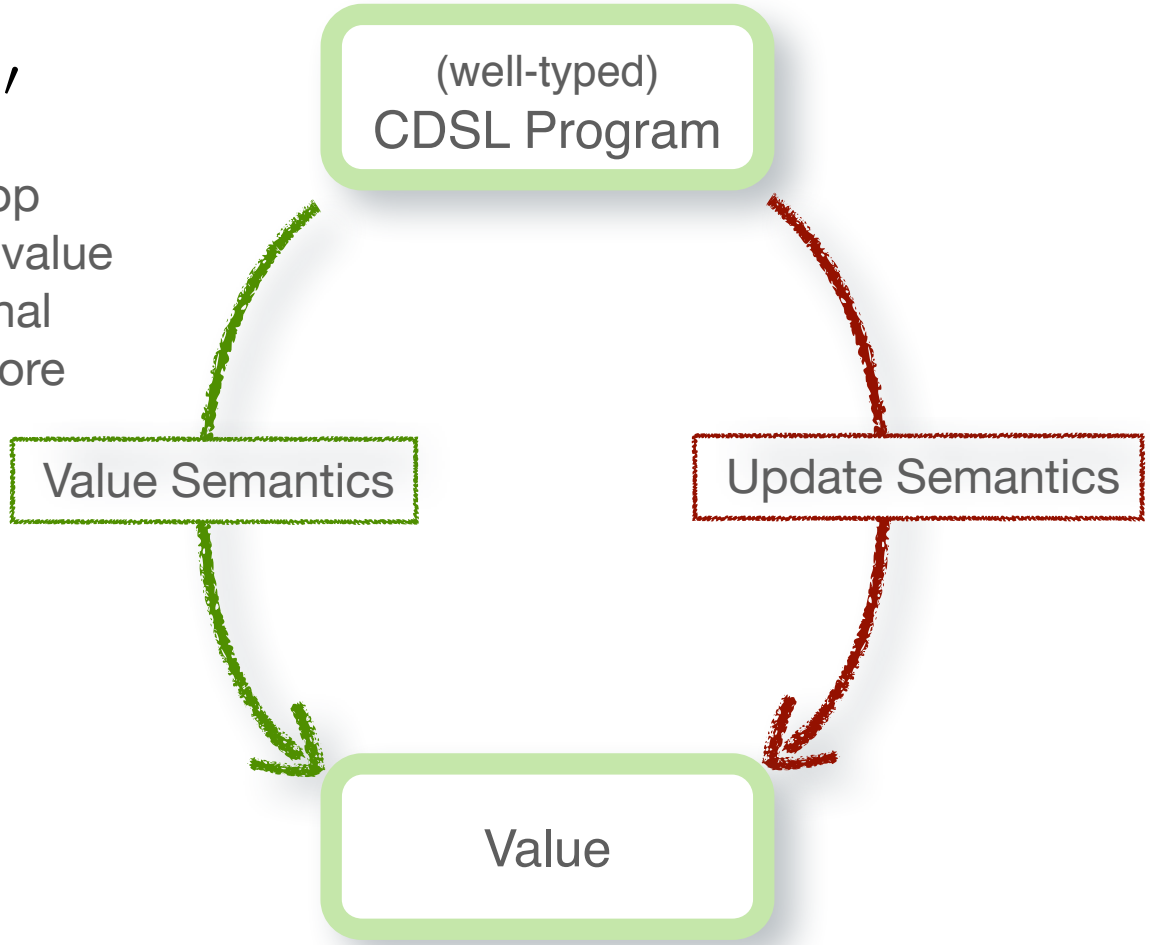


Value and Update Semantics of CDSL

NICTA

$$\Gamma \vdash e \mapsto e'$$

- `free` is a no-op
- Everything by value
- Purely functional
- No mutable store

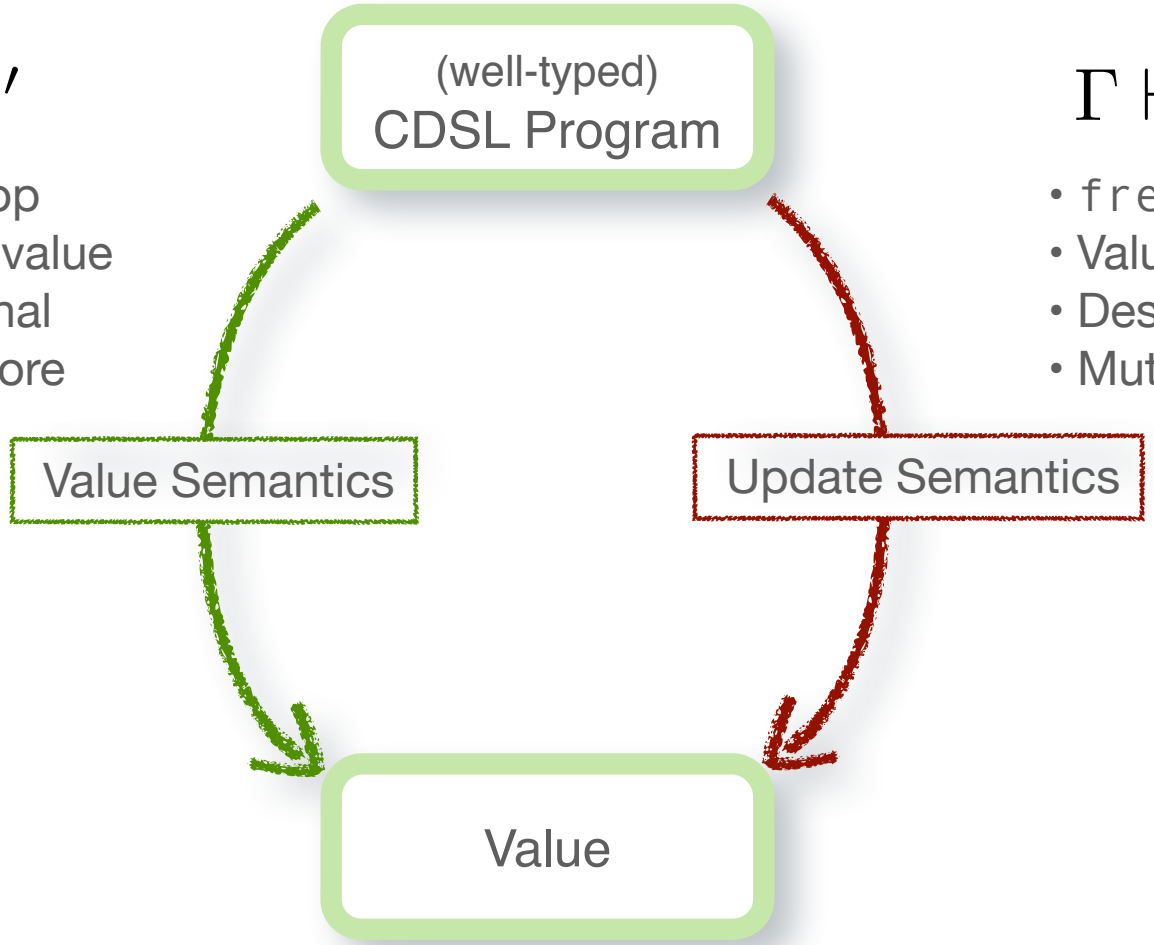


Value and Update Semantics of CDSL

NICTA

$\Gamma \vdash e \mapsto e'$

- `free` is a no-op
- Everything by value
- Purely functional
- No mutable store



$\Gamma \vdash \sigma, e \mapsto \sigma', e'$

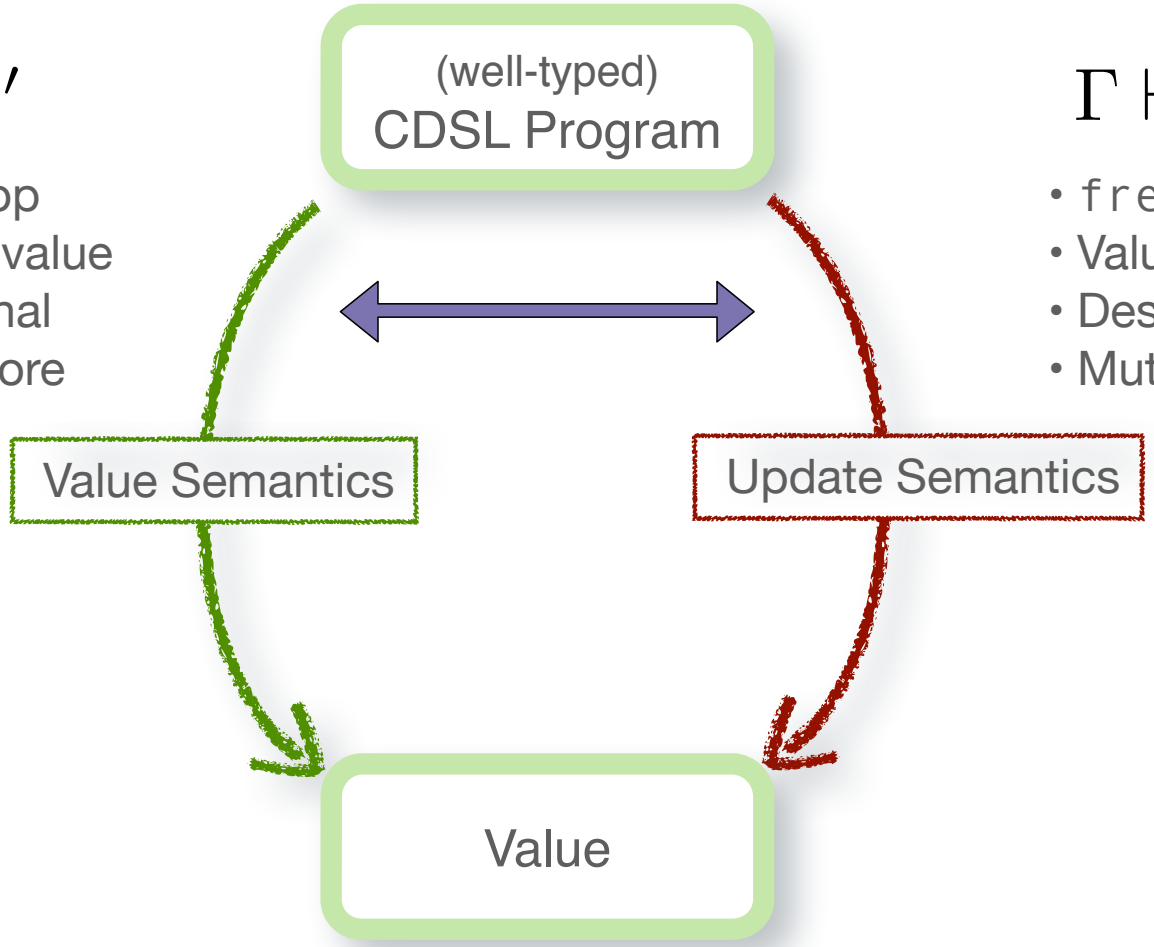
- `free` operates on a store
- Values can be store refs
- Destructive updates
- Mutable store

Value and Update Semantics of CDSL

NICTA

$\Gamma \vdash e \mapsto e'$

- `free` is a no-op
- Everything by value
- Purely functional
- No mutable store



$\Gamma \vdash \sigma, e \mapsto \sigma', e'$

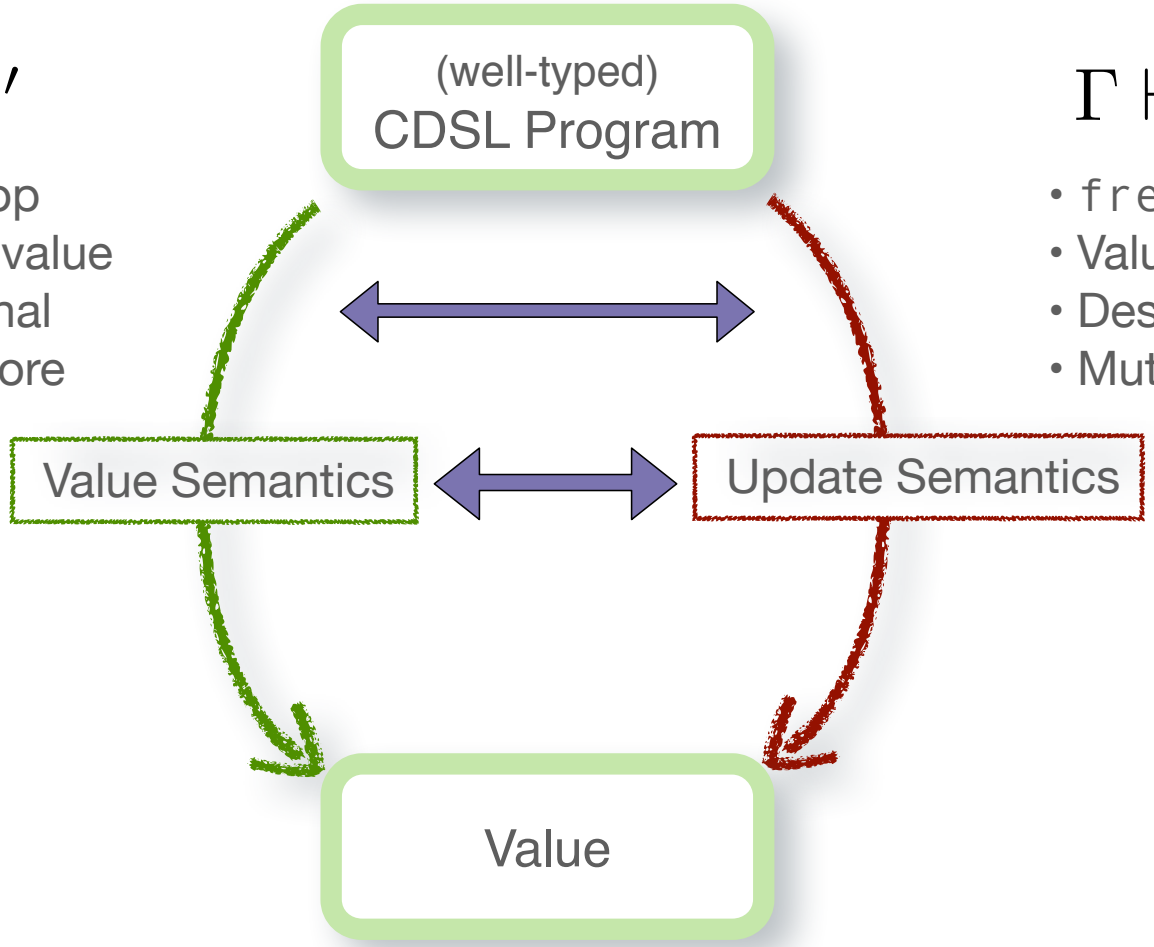
- `free` operates on a store
- Values can be store refs
- Destructive updates
- Mutable store

Value and Update Semantics of CDSL

NICTA

$\Gamma \vdash e \mapsto e'$

- `free` is a no-op
- Everything by value
- Purely functional
- No mutable store



$\Gamma \vdash \sigma, e \mapsto \sigma', e'$

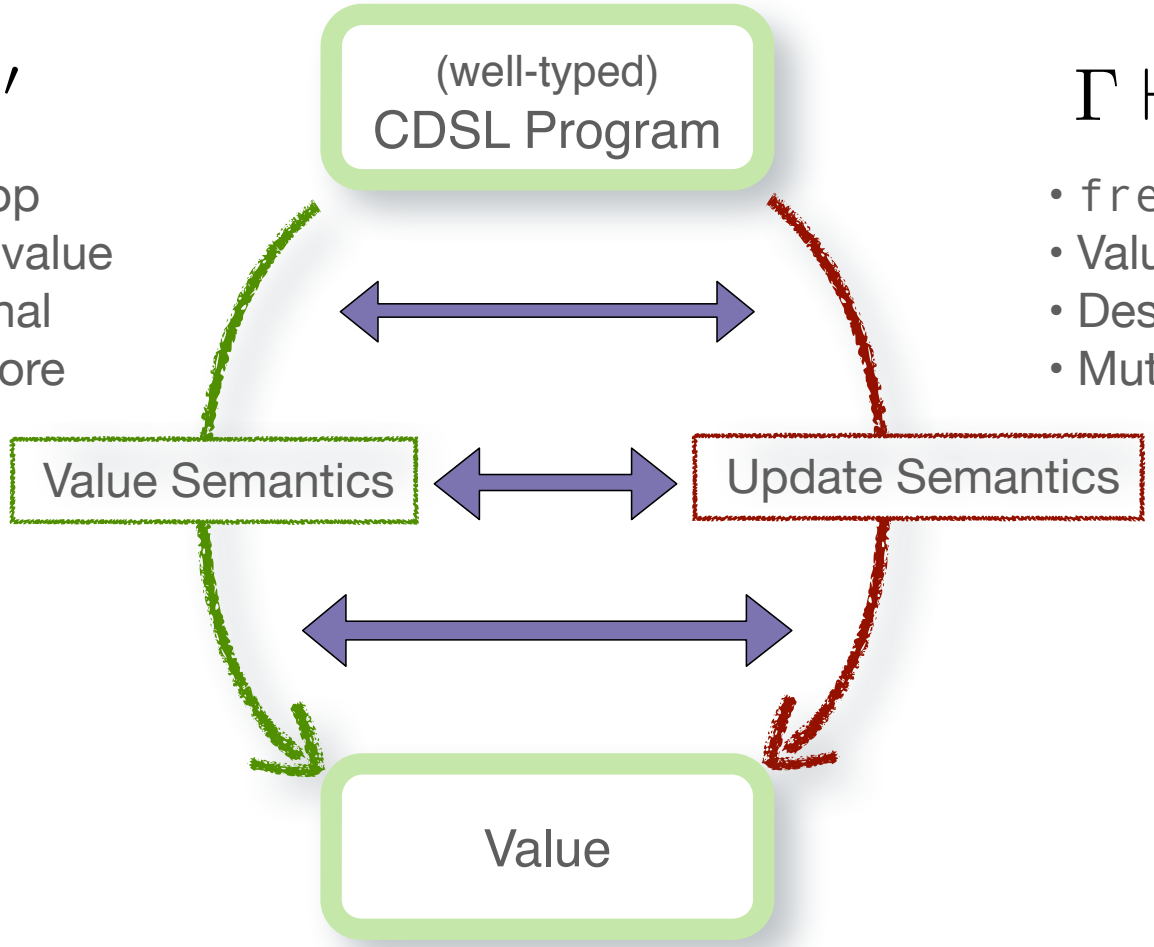
- `free` operates on a store
- Values can be store refs
- Destructive updates
- Mutable store

Value and Update Semantics of CDSL

NICTA

$\Gamma \vdash e \mapsto e'$

- `free` is a no-op
- Everything by value
- Purely functional
- No mutable store

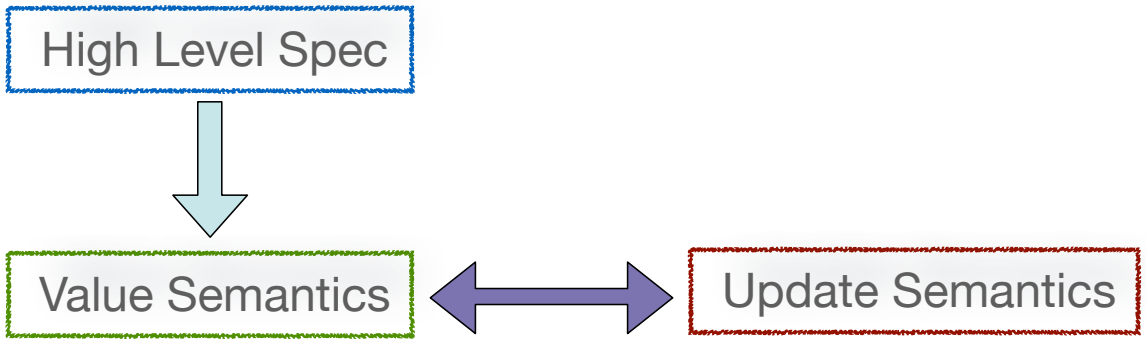


$\Gamma \vdash \sigma, e \mapsto \sigma', e'$

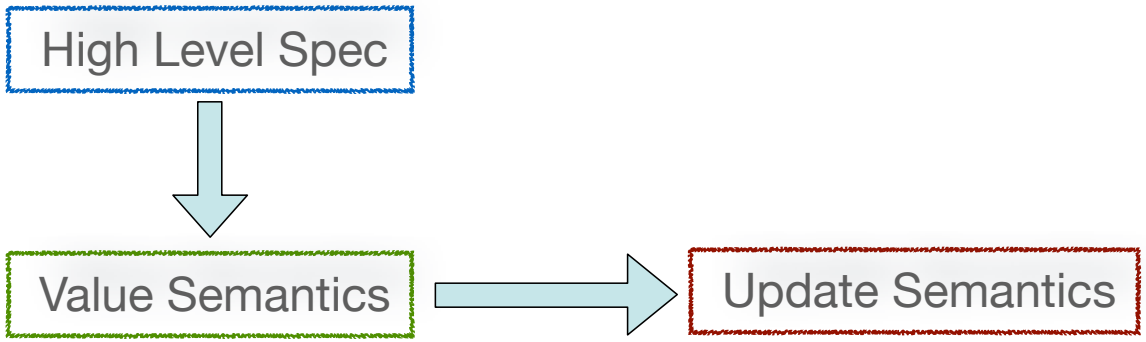
- `free` operates on a store
- Values can be store refs
- Destructive updates
- Mutable store

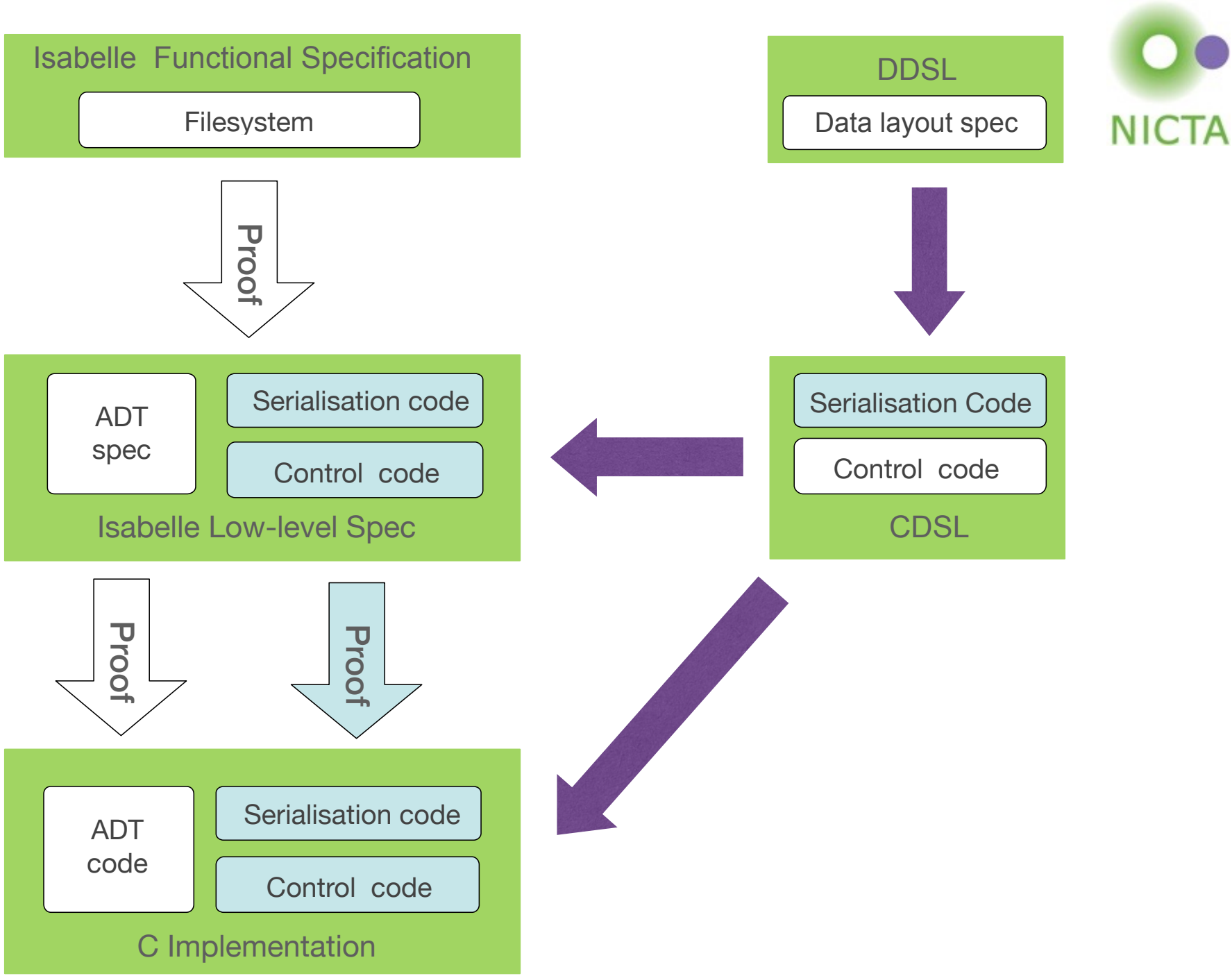
Value and Update Semantics of CDSL

Value and Update Semantics of CDSL



Value and Update Semantics of CDSL

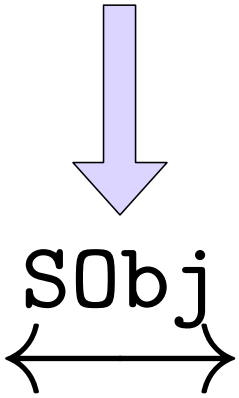



Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation


```
data SObj = {
   x :: Ple32,
   y :: U8
}
```

NICTA

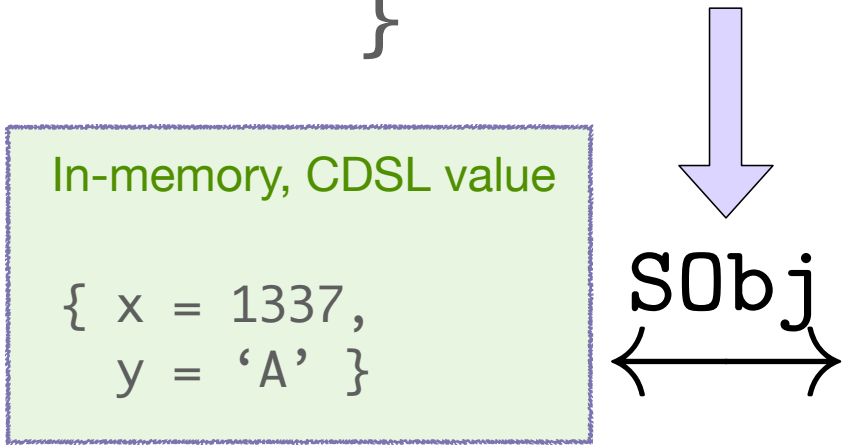
```
data SObj = {
    x :: Ple32,
    y :: U8
}
```



Design of the Data Description Language

NICTA

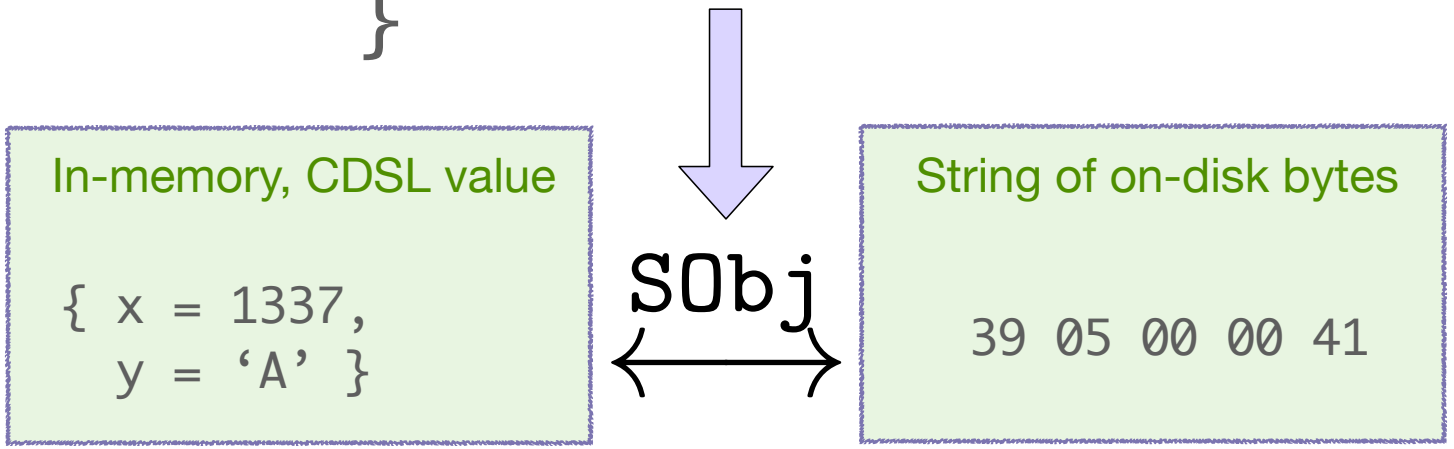
```
data SObj = {
    x :: Ple32,
    y :: U8
}
```



Design of the Data Description Language

NICTA

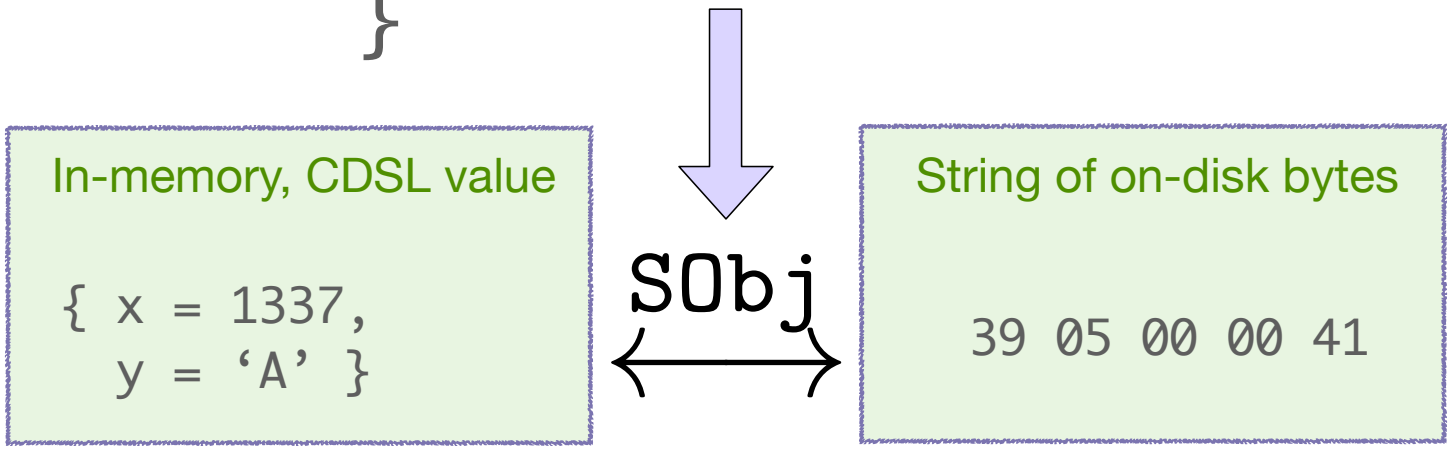
```
data SObj = {
    x :: Ple32,
    y :: U8
}
```



Design of the Data Description Language

NICTA

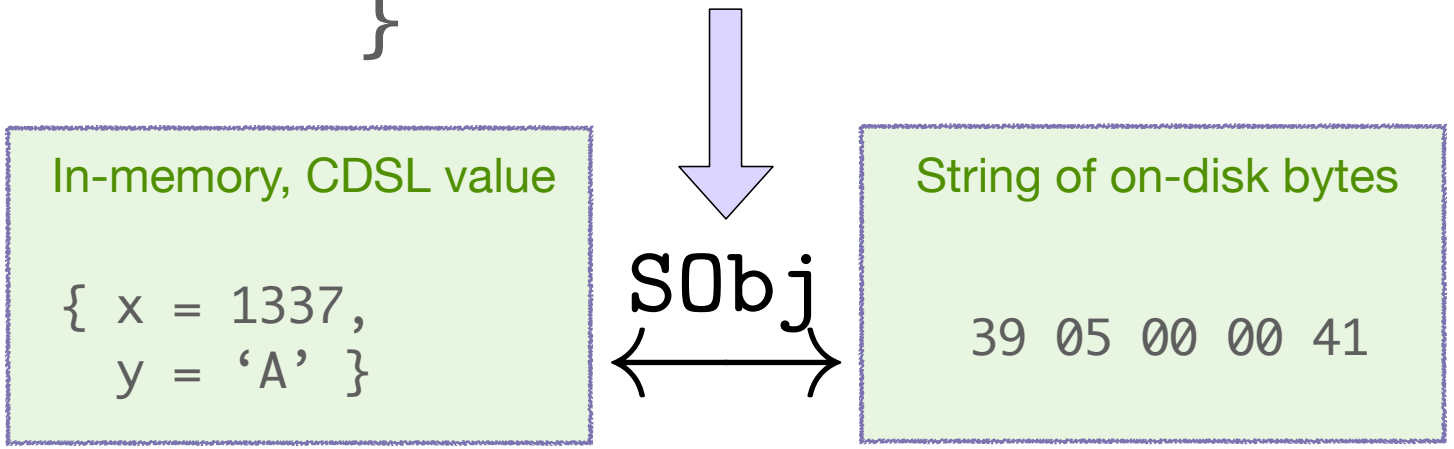
```
data SObj = {
    x :: Ple32,
    y :: U8
}
```



Design of the Data Description Language

NICTA

```
data SObj = {
    x :: Ple32,
    y :: U8
}
```



Design of the Data Description Language

NICTA

Existing DDLs

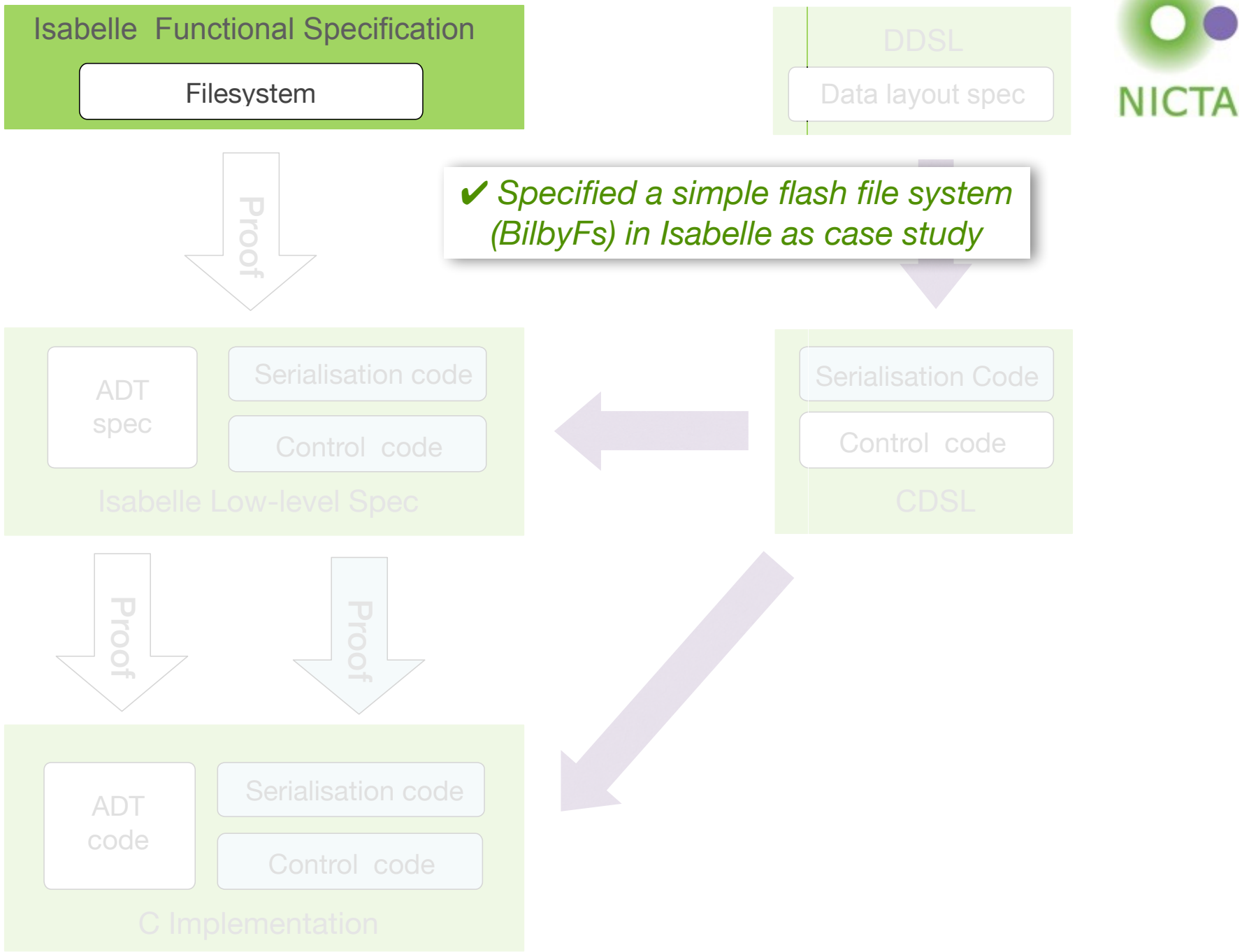
how to specify data structures and data layout

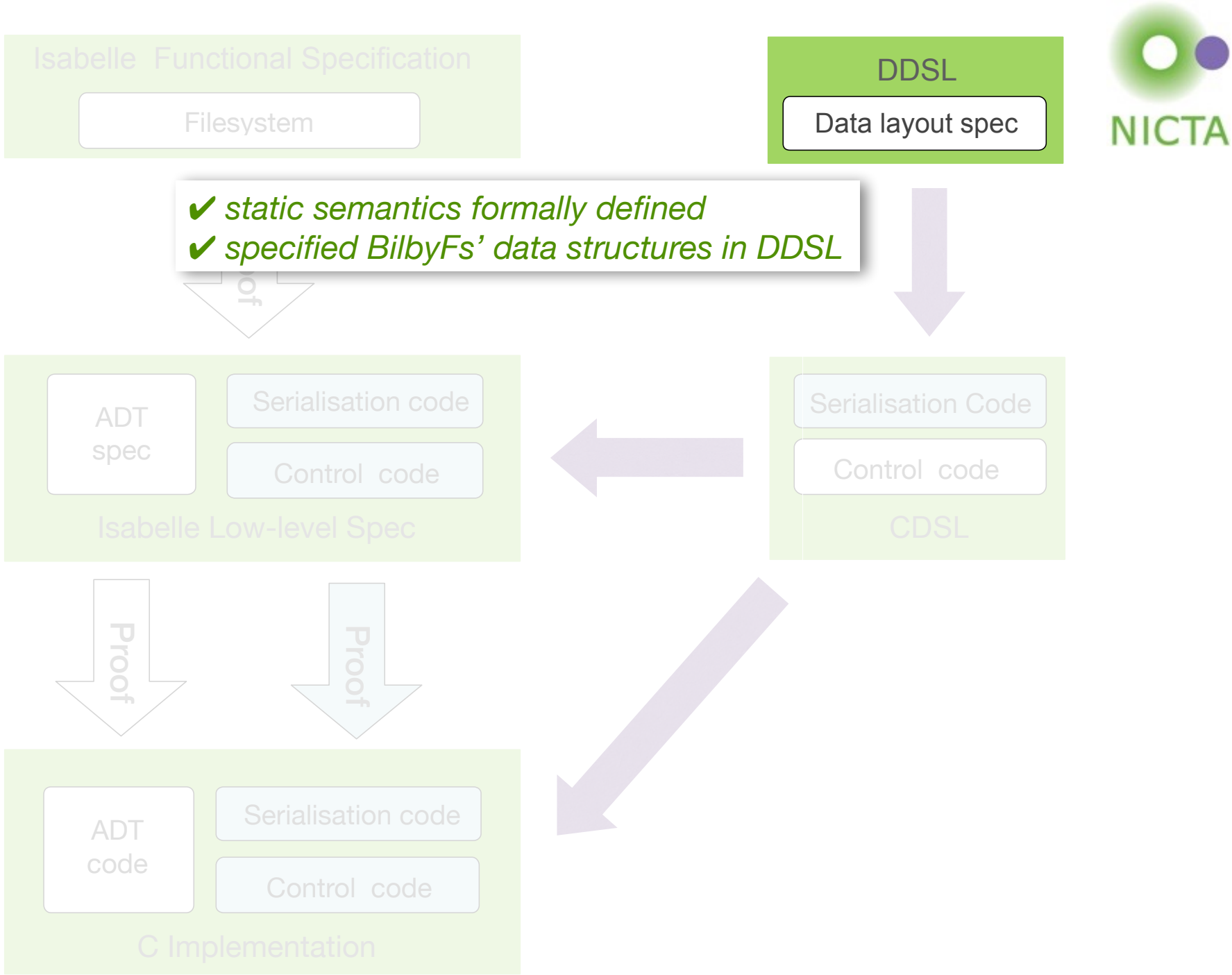
how to express constraints on the values of the data structures

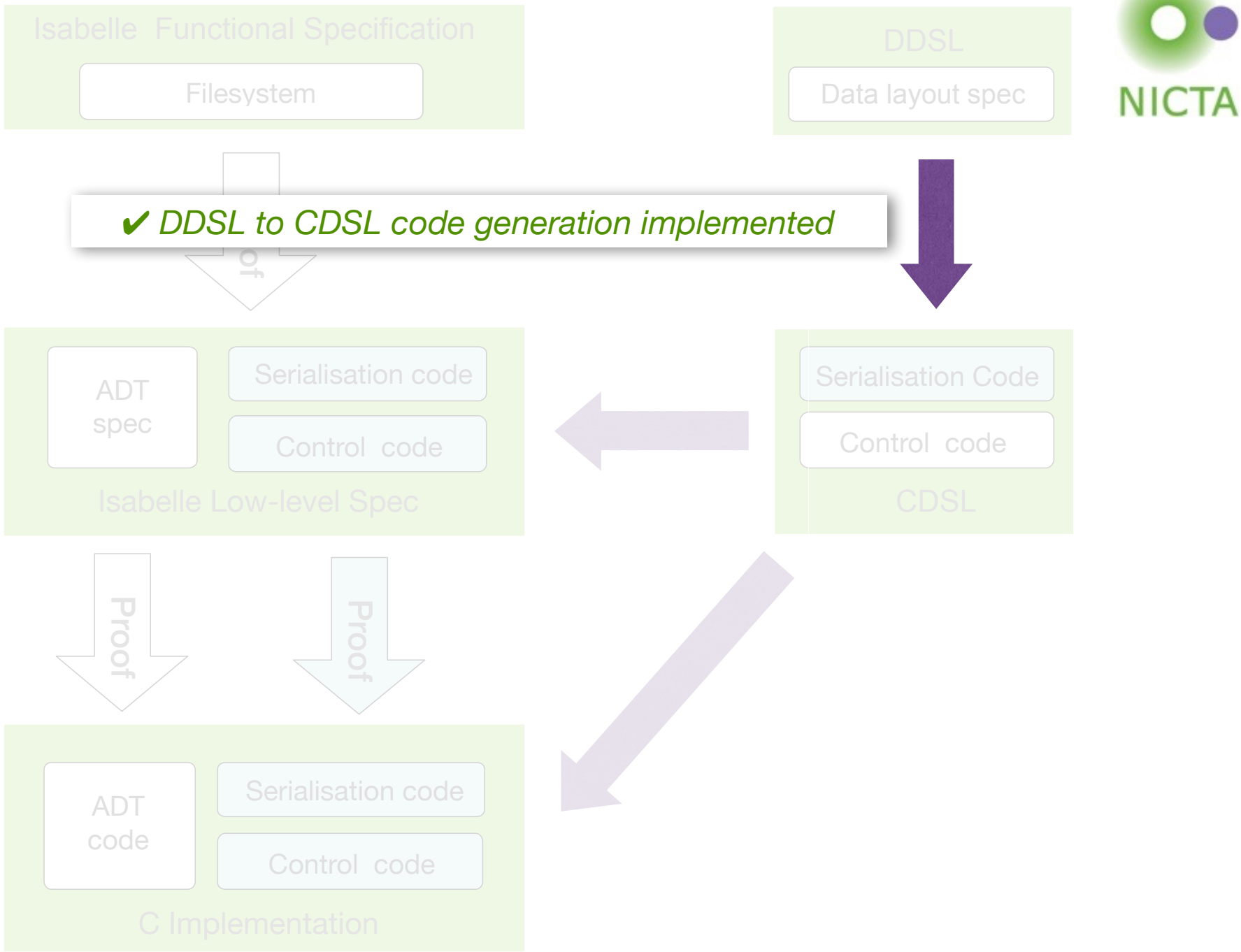
what and how to check statically

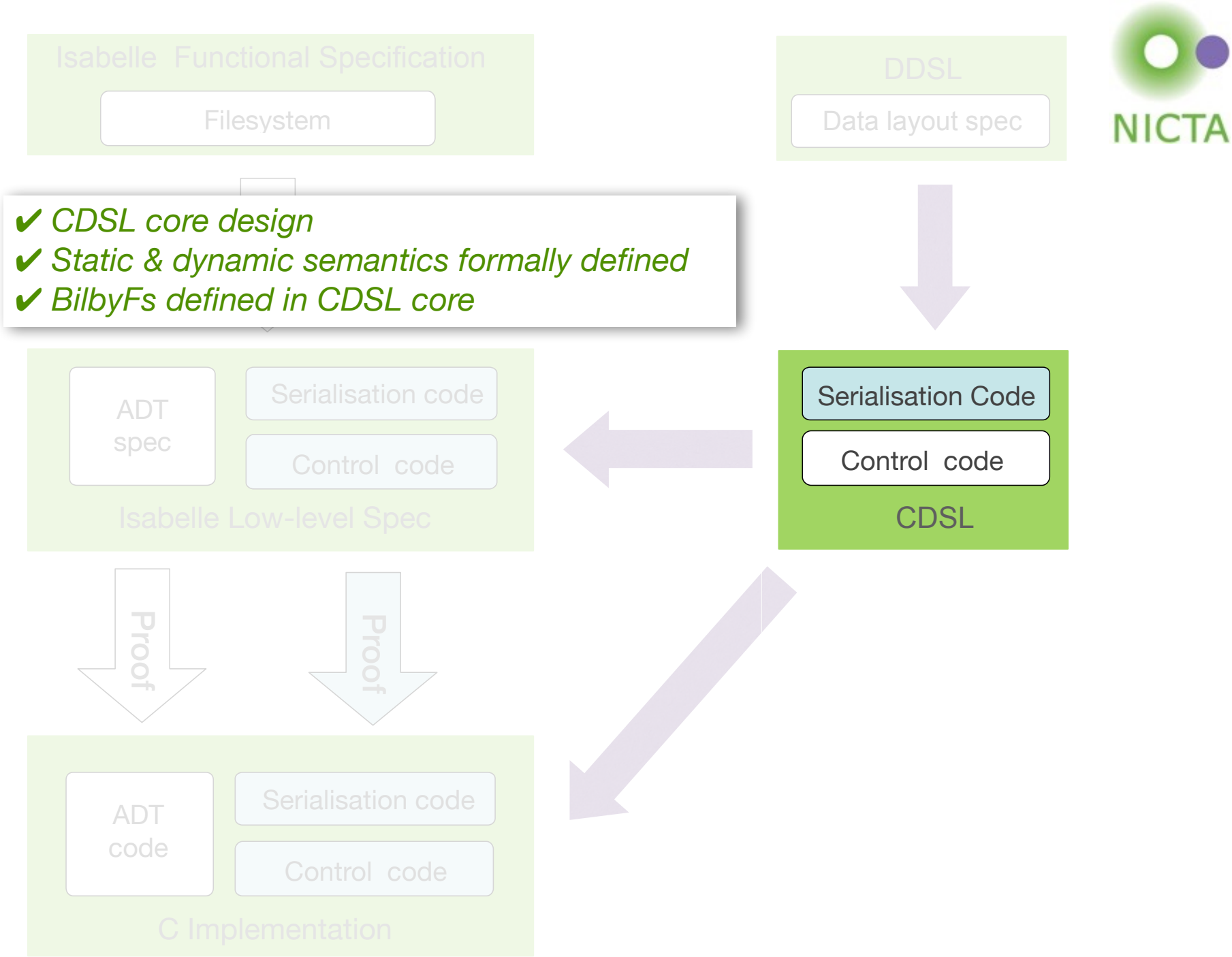
Existing verification tools

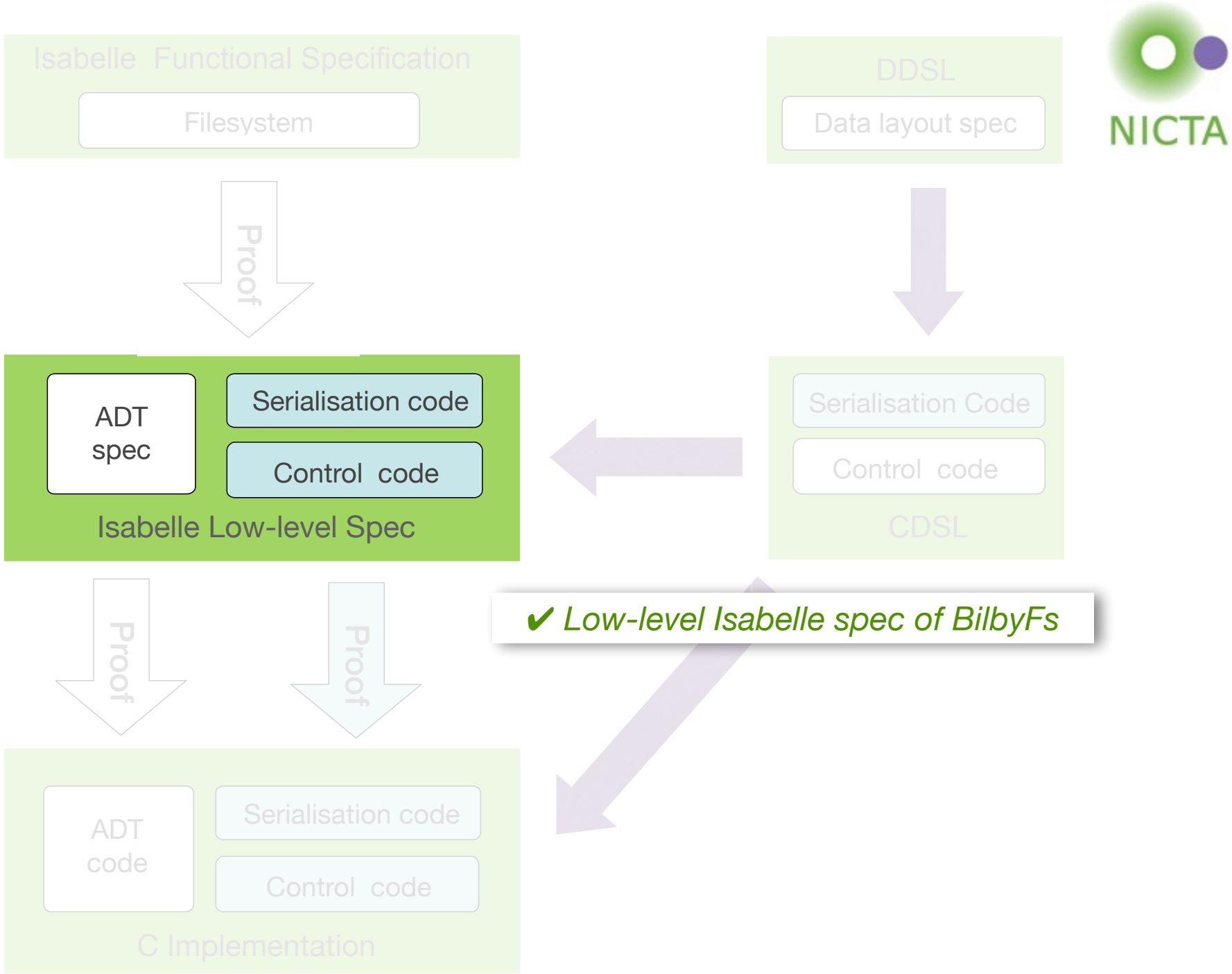
how to handle verification for tagged bitfield serialisation & de-serialisation

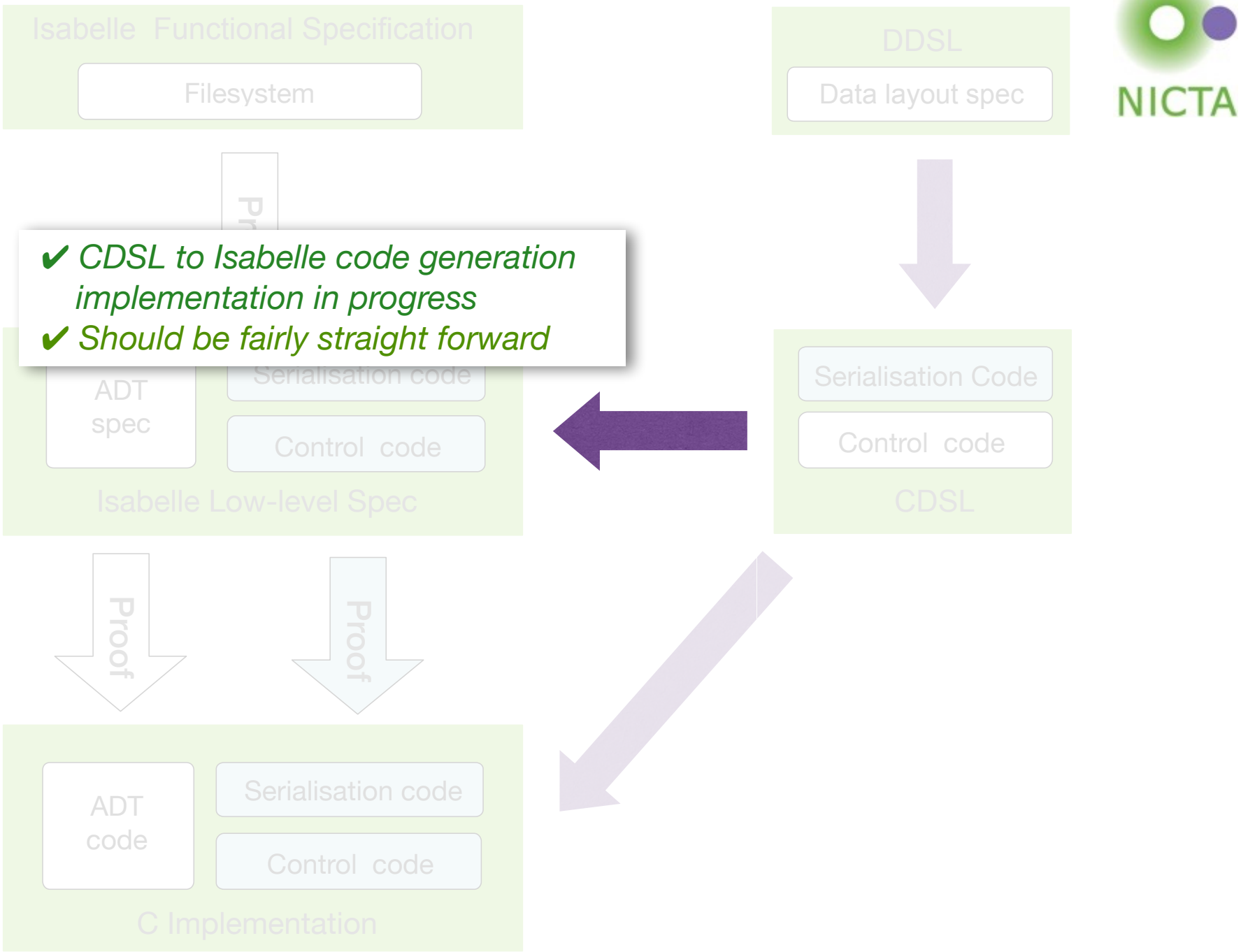

Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
Proof
✔ Specified a simple flash file system (BilbyFs) in Isabelle as case study
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation

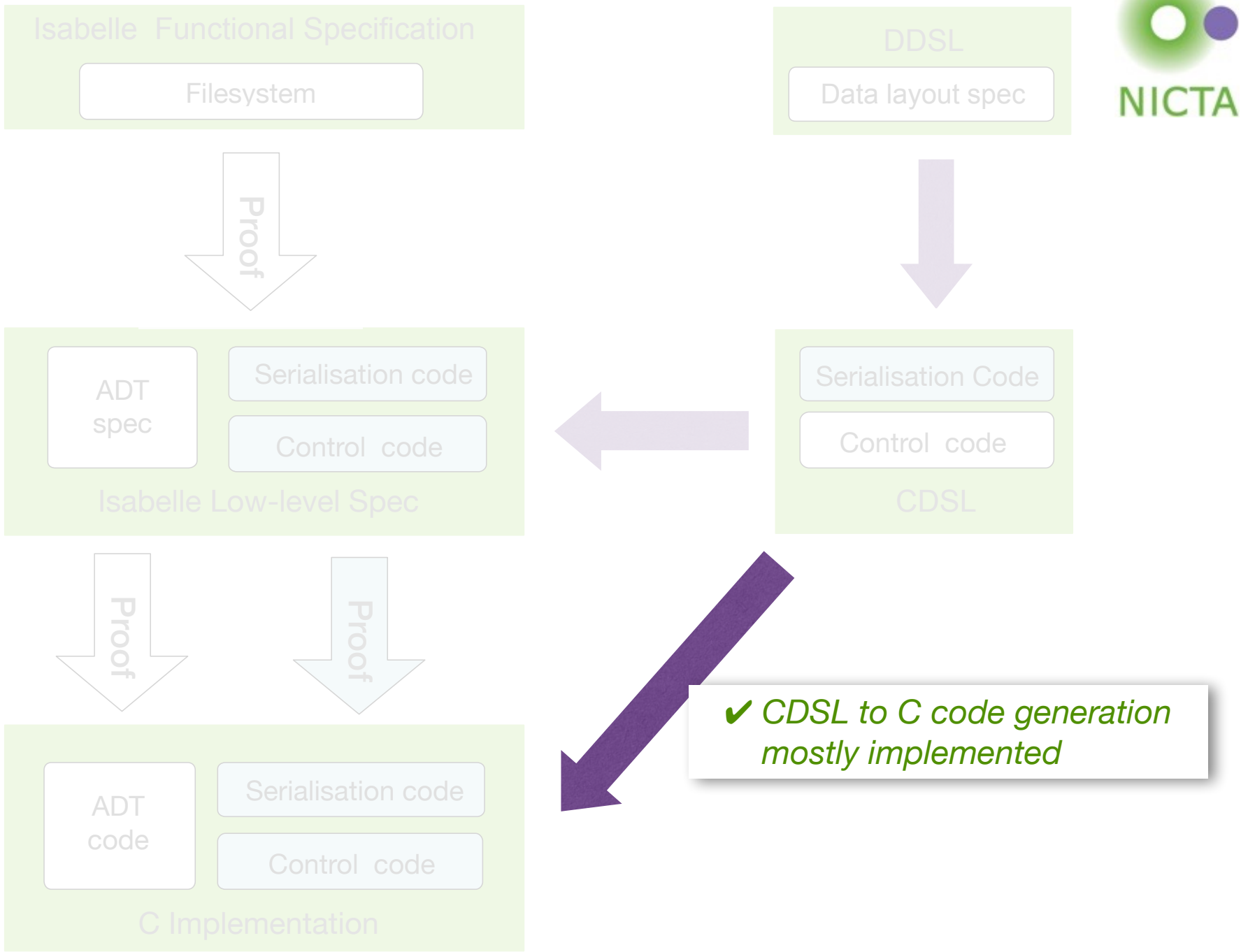

Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
✔ static semantics formally defined
✔ specified BilbyFs' data structures in DDSL
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation

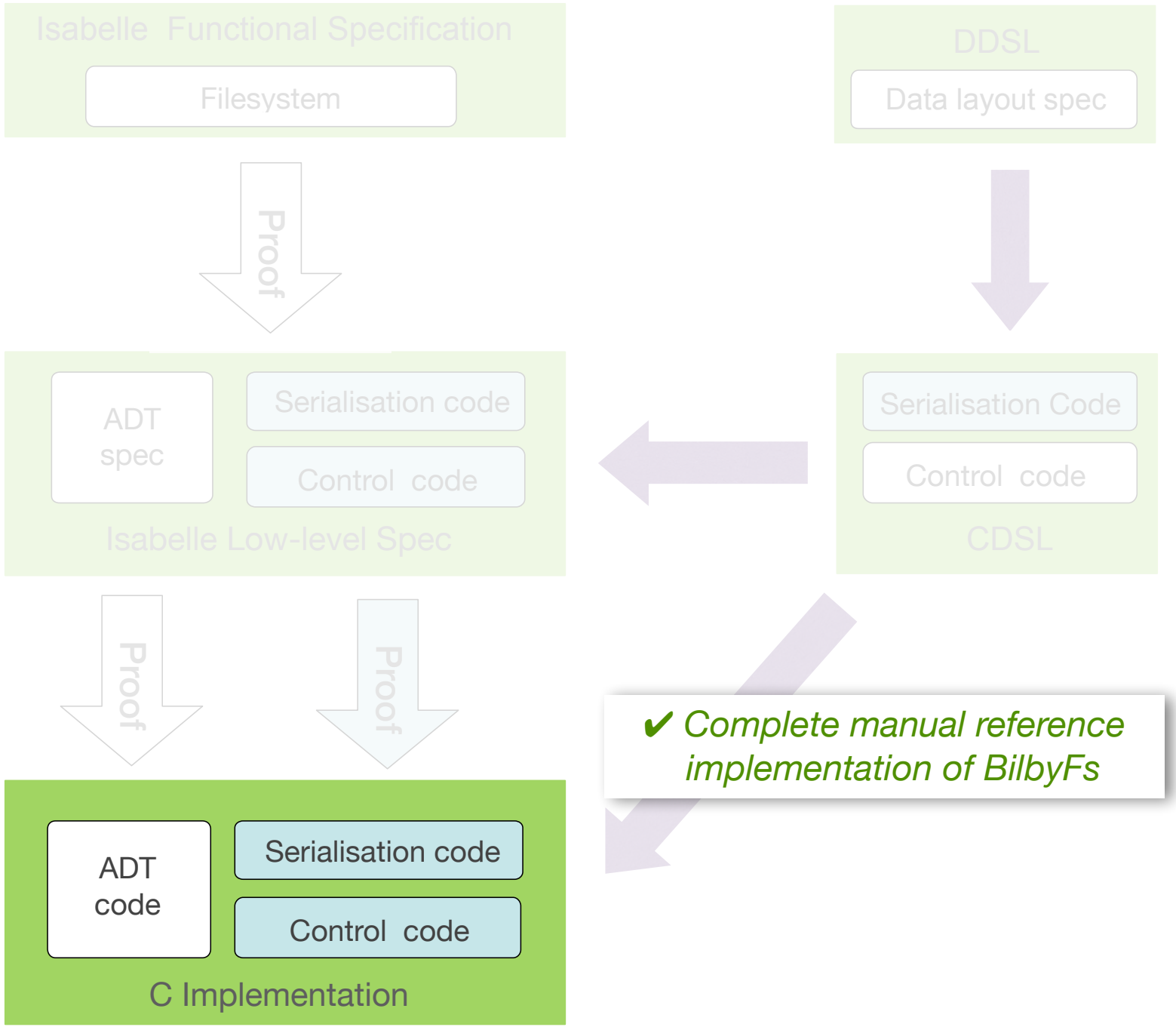

Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
✔ DDSL to CDSL code generation implemented
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation

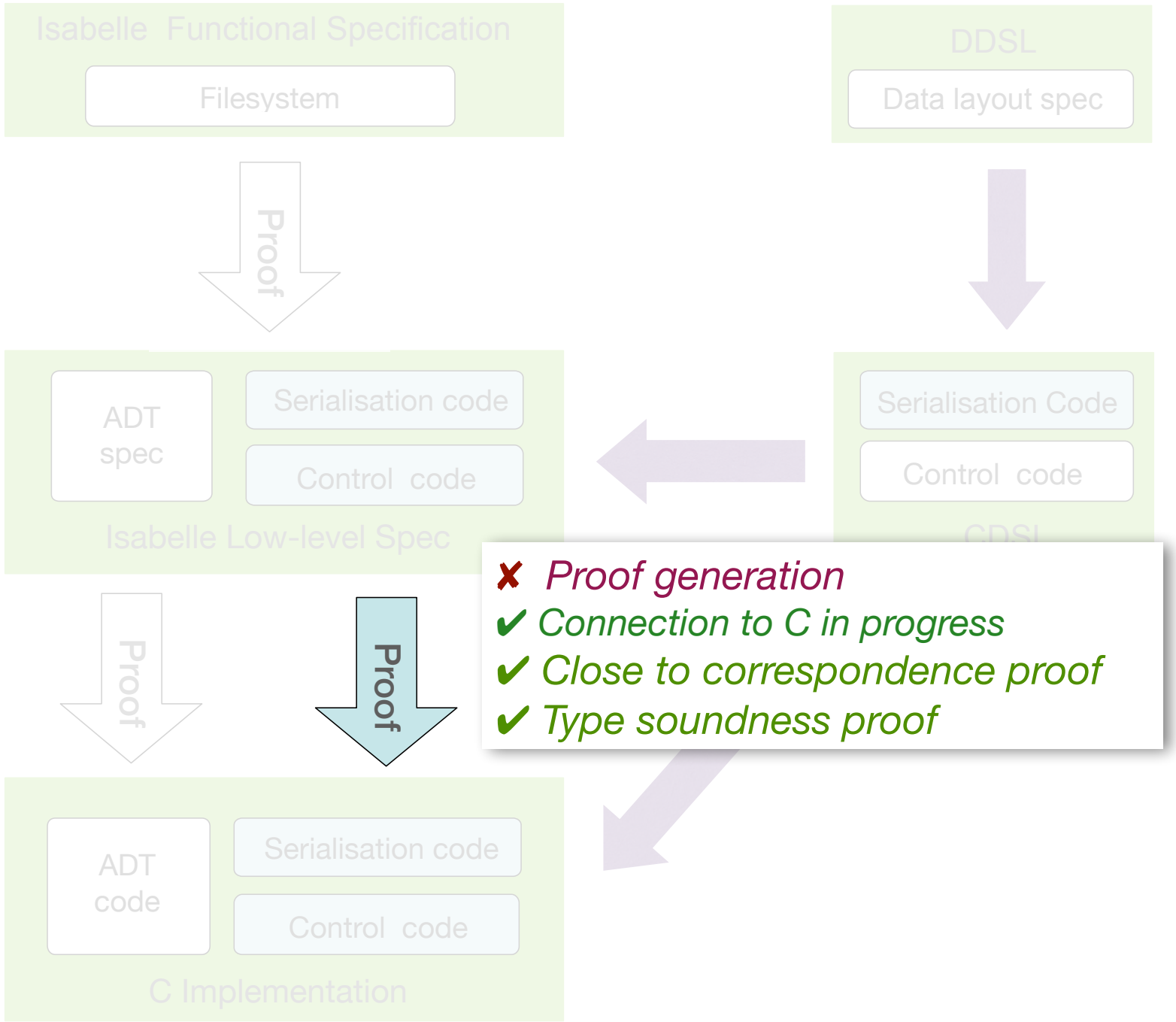

Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
✔ CDSL core design
✔ Static & dynamic semantics formally defined
✔ BilbyFs defined in CDSL core
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
✔ Low-level Isabelle spec of BilbyFs
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
✔ CDSL to Isabelle code generation implementation in progress
✔ Should be fairly straight forward
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
NICTA
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation
✔ CDSL to C code generation mostly implemented


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
CDSL
Proof
Proof
ADT code
Serialisation code
Control code
C Implementation
✔ Complete manual reference implementation of BilbyFs


Isabelle Functional Specification
Filesystem
DDSL
Data layout spec
Proof
ADT spec
Serialisation code
Control code
Isabelle Low-level Spec
Serialisation Code
Control code
DDSL
Proof
Proof
✘ Proof generation
✔ Connection to C in progress
✔ Close to correspondence proof
✔ Type soundness proof
ADT code
Serialisation code
Control code
C Implementation

Thank You