Data Management

Bein

1. Simplify calling external programs in your calculations

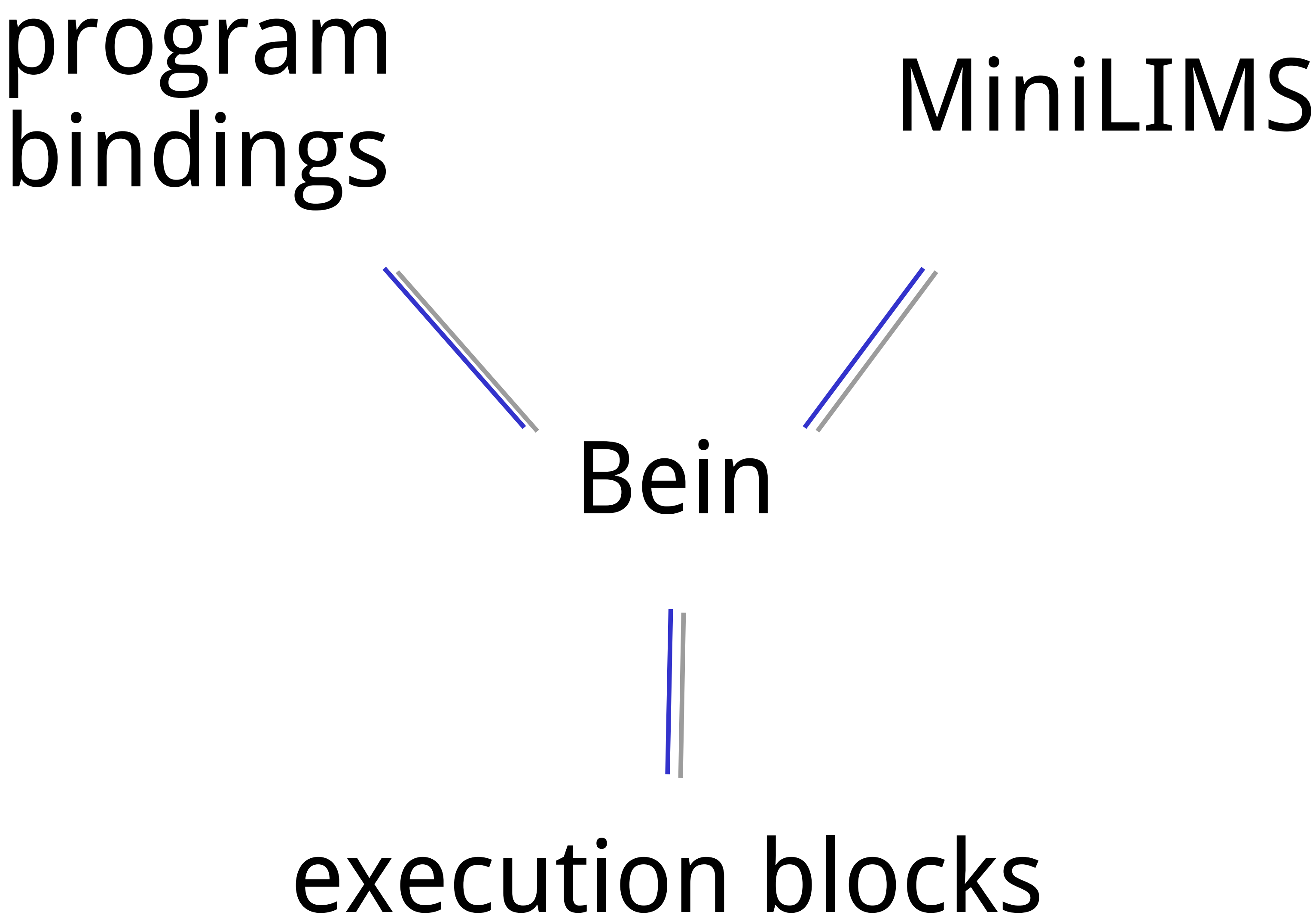
1. Simplify calling external programs in your calculations

2. Track the external progams you run and files to preserve

1. Simplify calling external programs in your calculations

2. Track the external progams you run and files to preserve

3. Minimize the distance between prototype and production code



program bindings

MiniLIMS

Bein

execution blocks

MiniLIMS

```
.../
    boris              ← SQLite database
    boris.files/       ← Directory for files
        ...
```

MiniLIMS

```
#!/usr/bin/env python
from bein import *


M = MiniLIMS(”/path/to/sqlitedb”)
```

Bein creates the database
if it does not exist

Execution Blocks

```
M = MiniLIMS(”/path/to/sqlitedb”)
with execution(M) as ex:
    ...
    ...

more_stuff()
```

Creates a working directory

Working directory cleaned up and deleted here automatically

Program Bindings

```
@program
def count_lines(filename):

  def parse_output(p):
    pat = r’^\s*(\d+)\s+’ + filename + r’\s*$’
    m = re.search(pat, ’’.join(p.stdout))
    return int(m.groups()[-1])

  return {”arguments”: [”wc”,”-l”,filename],
          ”return_value”: parse_output}
```

Program Bindings

Transform it into
a program binding

```
@program
def count_lines(filename):

  def parse_output(p):
    pat = r’^\s*(\d+)\s+’ + filename + r’\s*$’
    m = re.search(pat, ’’.join(p.stdout))
    return int(m.groups()[-1])

  return {”arguments”: [”wc”,”-l”,filename],
          ”return_value”: parse_output}
```

Program Bindings

Return arguments and a function to extract a return value

```
@program
def count_lines(filename):

  def parse_output(p):
    pat = r’^\s*(\d+)\s+’ + filename + r’\s*$’
    m = re.search(pat, ’’.join(p.stdout))
    return int(m.groups()[-1])

  return {”arguments”: [”wc”,”-l”,filename],
          ”return_value”: parse_output}
```

Program Bindings

```
with execution() as ex:
  n = count_lines(ex, filename)
  n = count_lines(ex, filename, stderr=“asdf”)
  f = count_lines.nonblocking(ex, filename)
  n = f.wait()
  f = count_lines.nonblocking(ex, filename,
                              via=“lsf”)
  n = f.wait()
```

```
#!/usr/bin/env python
import sys

a = f()
for i in range(5):
  ...
```

```
#!/usr/bin/env python
import sys
from bein import *

a = f()
for i in range(5):
    ...
```

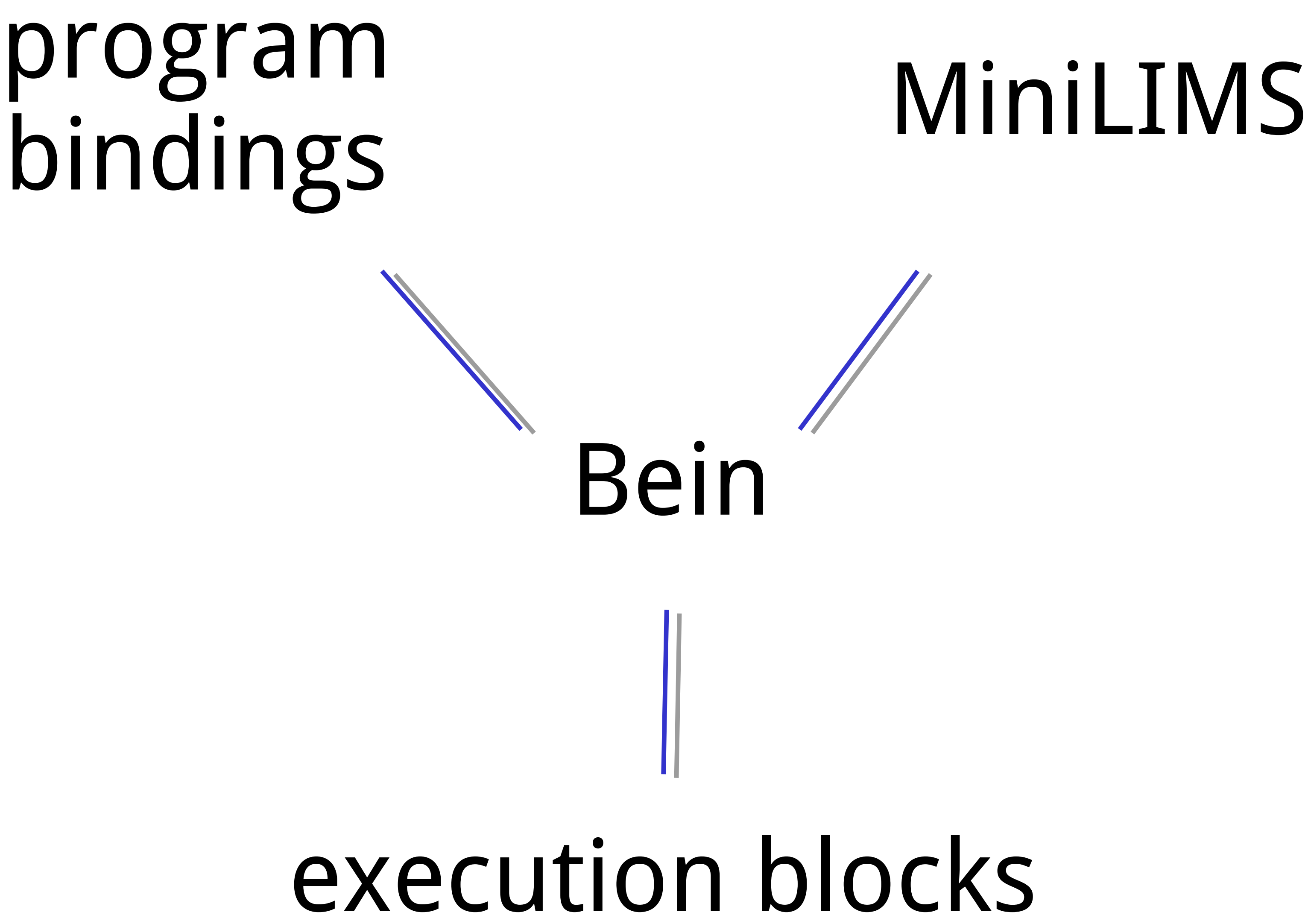
```
#!/usr/bin/env python
import sys
from bein import *

with execution() as ex:
  a = f()
  for i in range(5):
    ...
```

```
#!/usr/bin/env python
import sys
from bein import *

M = MiniLIMS(”/path/to/sqlitedb”)

with execution(M) as ex:
  a = f()
  for i in range(5):
    ...
```

pip install bein

Documentation at:

http://madhadron.com/bein/



program bindings

MiniLIMS

Bein

execution blocks

created by

Frederick Ross

March 2011