LIVV:

Design & Architecture

Andrew Bennett, Kate Evans, Joe Kennedy, Pat Worley

What is LIVV?

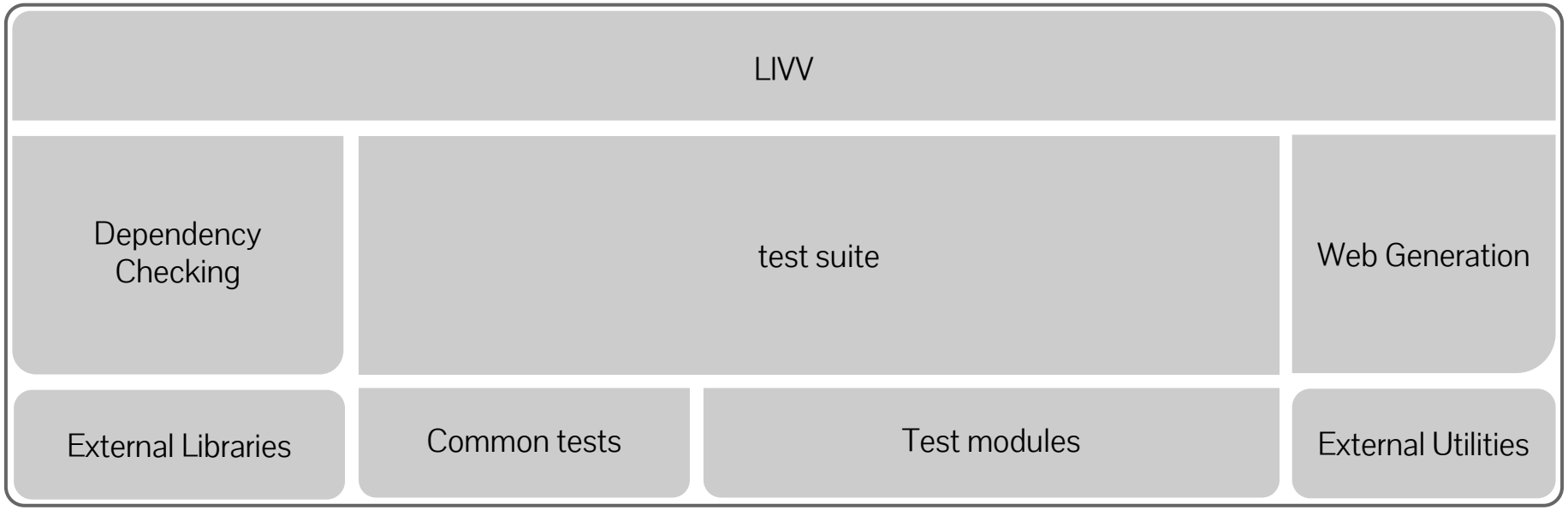
- **Land Ice** This one is easy.
- **Verification** Does the code work correctly?
- **Validation** Does the code do what we want it to?

Good Design = Good Tools

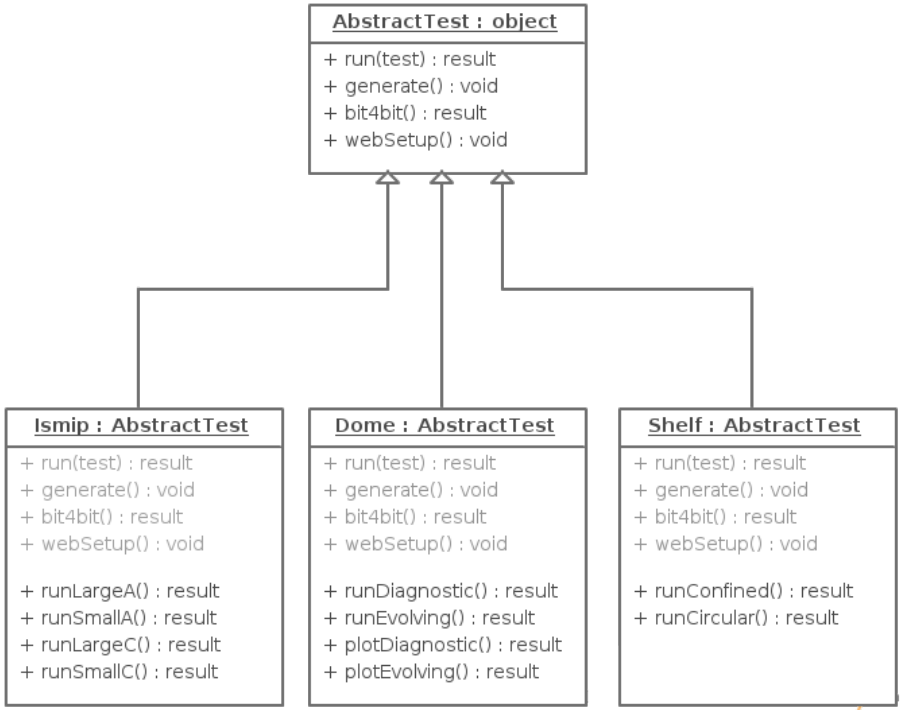
- Internal Dependency Management
- Leverage existing technologies
- Separate input, processing, and output

- Easy to set up and start using productively
- More productivity with less code
- Modular, extensible, and maintainable

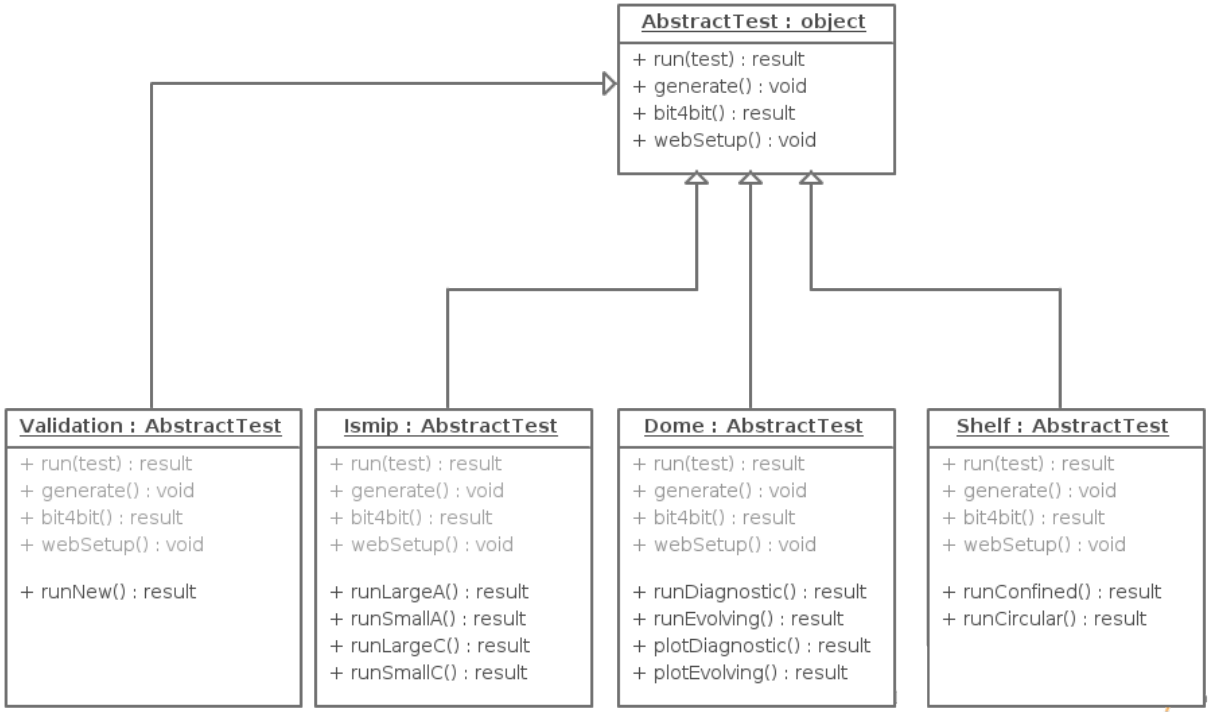
Design


LIVV
Dependency Checking
test suite
Web Generation
External Libraries
Common tests
Test modules
External Utilities

Test Suites


AbstractTest : object
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
Ismip : AbstractTest
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
+ runLargeA() : result
+ runSmallA() : result
+ runLargeC() : result
+ runSmallC() : result
Dome : AbstractTest
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
+ runDiagnostic() : result
+ runEvolving() : result
+ plotDiagnostic() : result
+ plotEvolving() : result
Shelf : AbstractTest
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
+ runConfined() : result
+ runCircular() : result

Test Suites


AbstractTest : object
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
Validation : AbstractTest
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
+ runNew() : result
Ismip : AbstractTest
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
+ runLargeA() : result
+ runSmallA() : result
+ runLargeC() : result
+ runSmallC() : result
Dome : AbstractTest
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
+ runDiagnostic() : result
+ runEvolving() : result
+ plotDiagnostic() : result
+ plotEvolving() : result
Shelf : AbstractTest
+ run(test) : result
+ generate() : void
+ bit4bit() : result
+ webSetup() : void
+ runConfined() : result
+ runCircular() : result

Good Design?

- The code worked fine before.
- Refactoring…? Boring.
- If it's so easy to add on where's the new stuff!?

New Features!

- Internal Dependency Management
- User specified configurations
- Simplified IO via templating

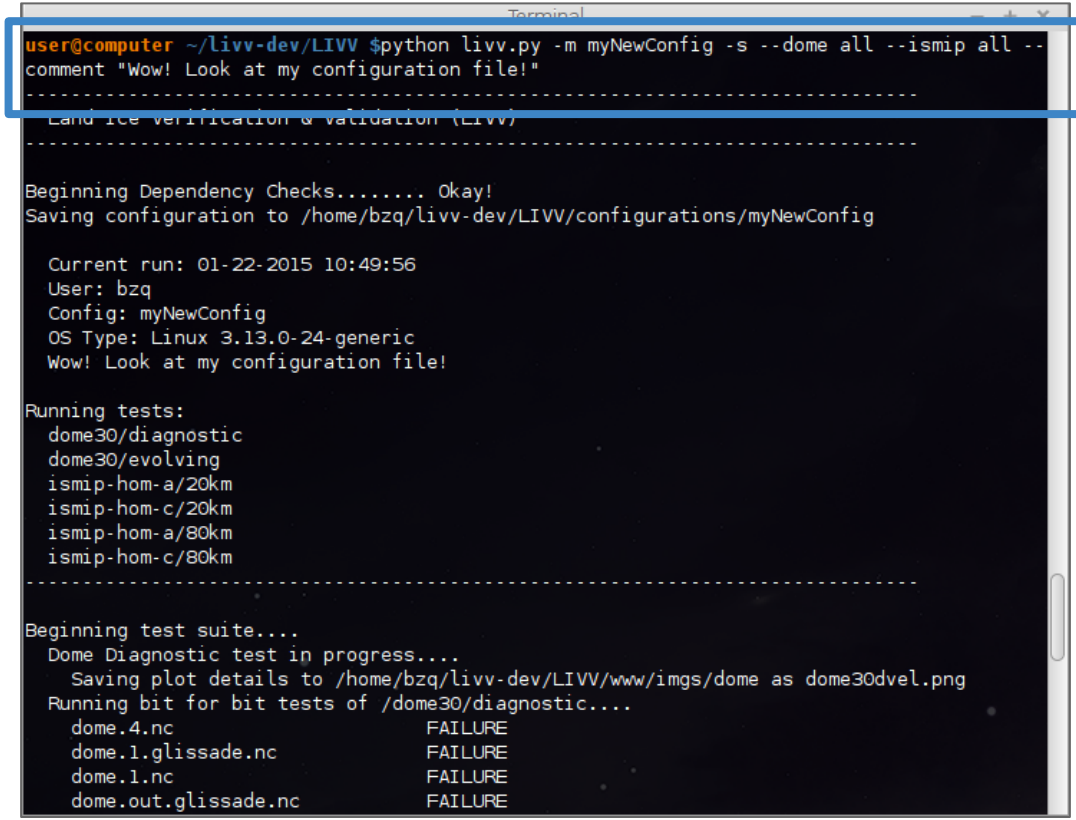
Dependency Management

- Nobody likes figuring out how to install software

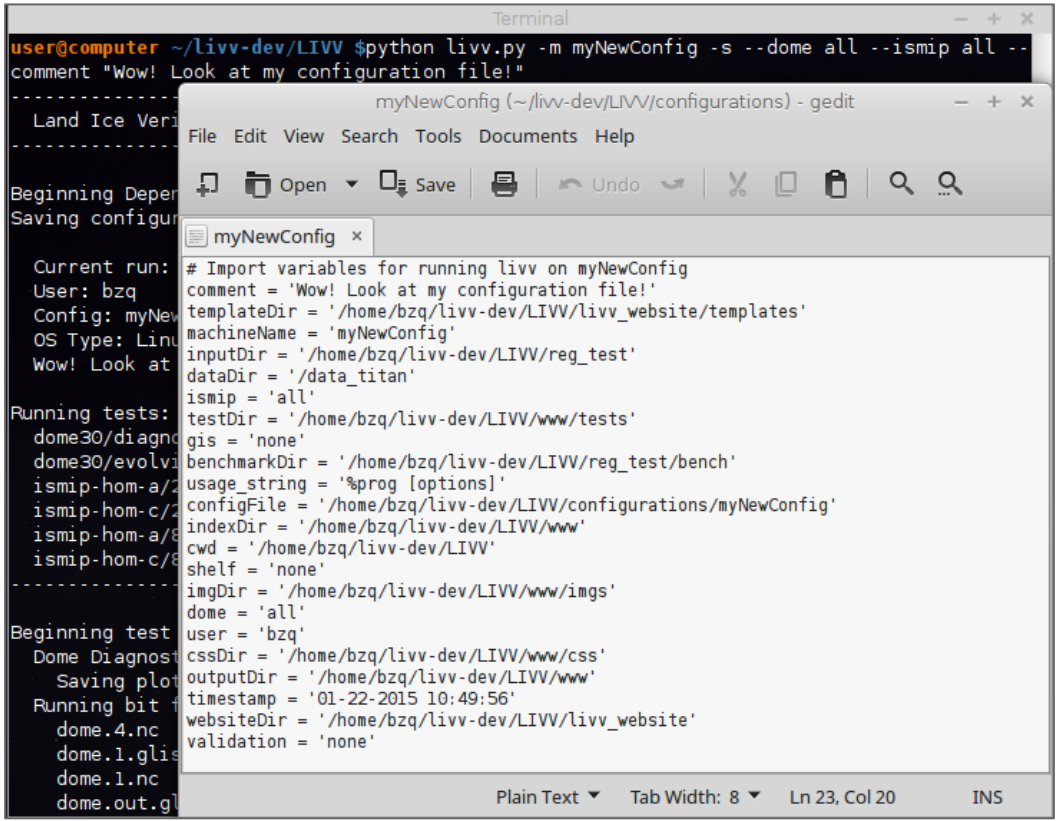
Dependency Management

- So far, limited to installing Python libraries
- Soon, installing system dependencies (NCL, NetCDF, etc.)

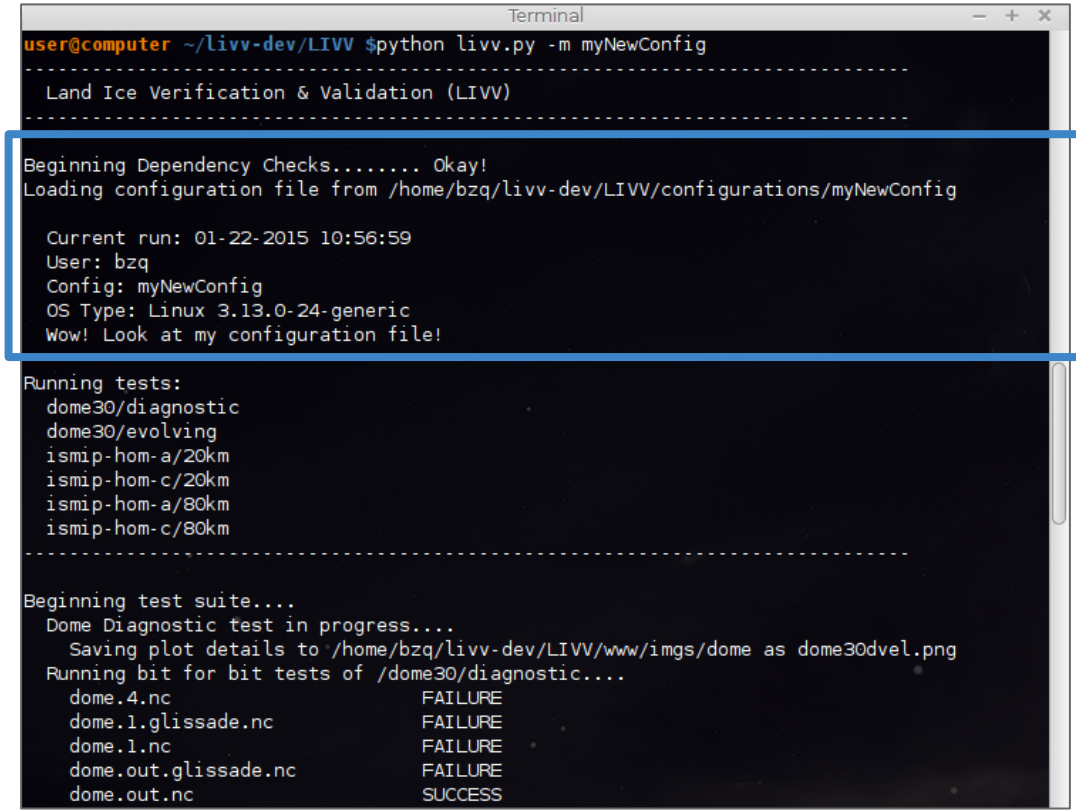
Configurations



Configurations

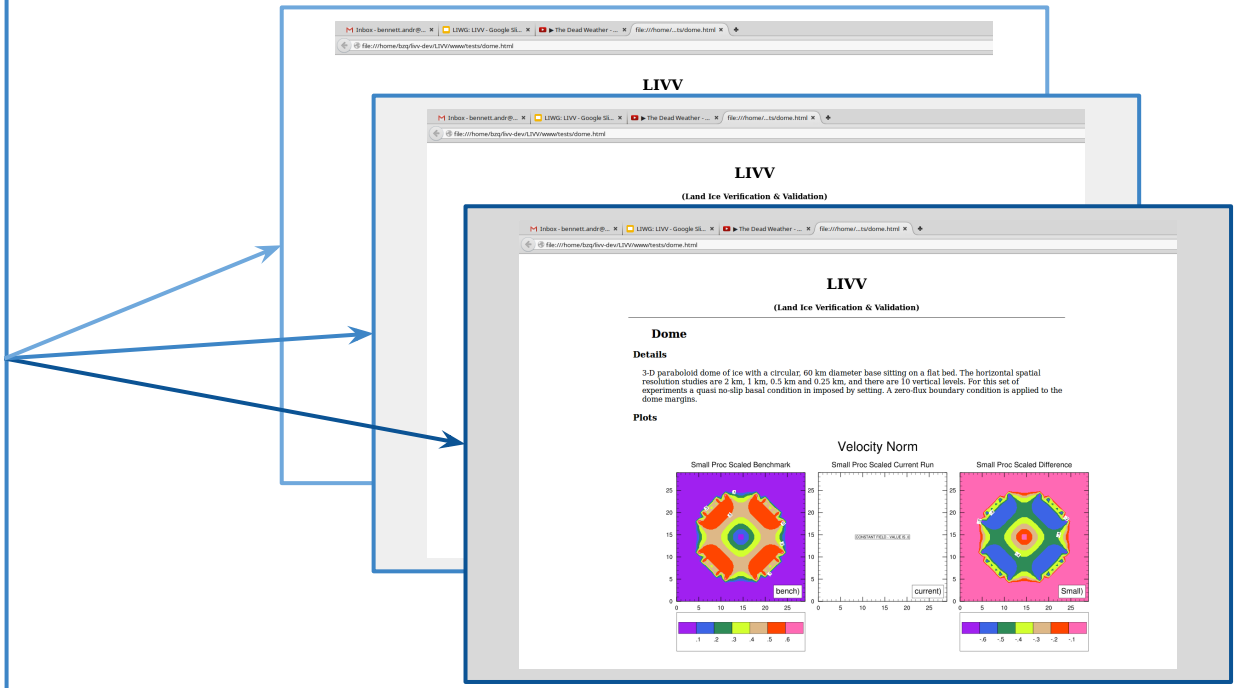


Configurations



Templating

```
<HTML>
  <HEAD>
    {% include 'header.html' %}

    <body>
      <h2>{{testName}}</h2>

      <h3>Details</h3>
      <p>
        {{testDescription}}
      </p>

      <h3>Plots</h3>
      <p>
        {% for img in testImages %}
          <img src="{{img}}" alt="{{img}}">
          <br>
        {% endfor %}
      </p>

      <h3>Results</h3>
        {% for case in testsRun %}
          <h4>{{case}}</h4>
          <p id="bold"> Bit for Bit </p>
            <p>Bit for bit results of {{case}}.</p>
          <p id="bold"> Standalone </p>
            <p>Results for {{case}} go here.</p>
        {% endfor %}
      <a href="{{indexDir}}/index.html">Home</a>
    </body>

    {% include 'footer.html' %}
  </HEAD>
</HTML>
```



The Future

- Support for additional dycores
- Upgrades to performance testing
- Comparison of models and observational data
- Tighter integration with CISM