



Integrating domain-specific package managers with distribution package management systems

Michael Homer

Wellington, January 18 2010

Domain-specific package managers

- RubyGems
- LuaRocks
- CPAN
- PEAR
- ...

Integration

There is none.

But there should be.

Separation

- Make use of the systems.
- But give them their own separate trees.
- Don't allow them to interact with or damage the base system.

Integration again

Allow a system package to have a dependency on a third-party one:

```
LuaRocks:json >= 1.0
```

Alternatives

- Wrap
- Abstain
- Endorse and mitigate

Interface

All of the form: `Alien AlienType:alienpkg mode [...]`

Three main modes:

- `--met min [max]`: Is the package installed?
- `--getinstallversion min [max]`: What version would be installed?
- `--install [version]`: Do it.

Examples

```
Alien --met LuaRocks:json 1.0 2
```

⇒ **exit 1**

```
Alien --getinstallversion LuaRocks:json 1.0 2
```

⇒ **output 1.2**

```
Alien --install LuaRocks:json 1.2
```

⇒ **Do it.**

Wrapper examples

```
Alien-LuaRocks --met json 1.0 2
```

⇒ **exit 1**

```
Alien-LuaRocks --getinstallversion json 1.0 2
```

⇒ **output 1.2**

```
Alien-LuaRocks --install json 1.2
```

⇒ **Do it.**

Limitations

There are two big ones.

Reverse dependencies

- An open question.

Summary

- Allow users to use the third-party packaging systems they like.
- Also tie them into the system.
- Try not to break anything.