Towards a Web-based Analyzer to improve the teaching of Alloy

Alcino Cunha, Nuno Macedo
José Pereira, Bruno Carvalho, Ricardo Silva
I teach Alloy at University of Minho since 2007

I love Alloy because of its simplicity, focus on abstraction, and polished analyzer

But every year many students struggle with it, and we need to understand why

We had some success using auto-grading systems in introductory programming courses; could such a system be beneficial for learning Alloy?

Could a Pex4Fun like web-based environment help spread the Alloy mania?
Requirements

• Online model development

• Easy sharing of models and instances (preserving layout)

• Creation of specification challenges with automatic grading

• Collection of data for posterior mining to understand how students learn Alloy and their main obstacles in doing so

• Lightweight and anonymous (no accounts, no passwords, etc)

• Standard syntax to allow easy copy and paste from the web version to the standalone one
Design proposal

```plaintext
//LOCKED
sig A {
  r : set B
}
//LOCKED
sig B {

pred injective { // write a constraint forcing r to be injective
}

//SECRET
pred solution { all b : B | lone r.b }
//SECRET
check correctness { injective iff solution }
```
Design proposal

sig Model {
    derivationOf : lone Model
}
sig Command {
    model : one Model
}
sig Instance {
    command : one Command
}
abstract sig Link {
    linksTo : one Model+Instance
}
sig Public, Private extends Link {}
Design proposal

// The stored models form a forest of derivations
all m : Model | m not in m.^derivationOf
// You can only have private links to models
all p : Private | p.linksTo in Model
// There is at most one public and one private link for each artefact
all a : Model+Instance | lone linksTo.a & Public and
                   lone linksTo.a & Private
// Sharing a model with new secrets creates public and private links
all m : Model | some linksTo.m & Private implies
                     some linksTo.m & Public
// Secrets can only be introduced on models without secrets
all p : Private | some p.linksTo.derivationOf implies
                     no linksTo.(p.linksTo.^derivationOf) & Private
// A model is also stored when running a command,
// but since it was not shared it can only be derived once
all m : Model | no linksTo.m implies
                    some model.m and lone derivationOf.m
// Instances are only stored if shared
all i : Instance | some linksTo.i
Example

Diagram of model relationships:
- Public2 linksTo Model1
- Private derivationOf Model1
- Model1 derivationOf Model2
- Public1 linksTo Model1
- Model1 derivationOf Model3
- Model3 model Command1
- Command0 model Command1
- Command0 linksTo Instance
- Instance linksTo Public0
Discussion

• Can the current design be improved?
  • Is the granularity (paragraph) of secrets and locked elements adequate? What other data should be collected?

• Am I being too stubborn about the requirements?
  • Could accounts be useful? Could specialised syntax improve the usability?

• Who wants to join this project?
  • Improving the design and architecture, developing, hosting, preparing and running challenges and studies, etc