


June 2-3, 2014, Toulouse
5th Rodin Workshop



Developing and Proving a Complicated System Model with Rodin

 Alexey Khoroshilov,
Ilya Shchepetkov
{khoroshilov,shchepetkov}@ispras.ru

ISPRAS

Institute for System Programming of the Russian Academy of Sciences

Statistics on the Model

	Number	Lines of code
Contexts	1	57
Sets	9	10
Constants	22	7
Axioms	18	39
Machines	1	1669
Variables	37	1
Sets	7	1
Functions	30	1
Invariants	100	288
Type invariants	37	45
State invariants	63	243
Events	35	1375
The largest event	-	241
The smallest event	-	8
An average event	-	40
Proof obligations	1226	-

Plugins

- ProB
- Camille
- Atelier B Provers
- SMT Solvers
(VeriT, CVC4 and Z3)



Some difficulties that we faced

Difficulties in developing the model:

- Complicated predicates
- Limitations of text editors

Difficulties in proving the model:

- Impossibility of team verification
- Excessive number of automatically added hypotheses
- Too much time spent on auto proving
- Auto tactics



Complicated predicates

Problem: the following construction, for example, is duplicated in many places of our model:

$$\begin{aligned} & \text{Downgrade} \mapsto \text{ReadA} \in \text{SessionCapabilities}(\text{session}) \\ & \vee \\ & (\text{EntitySkILevel}(\text{container}) = \text{SessionSkILevel}(\text{session}) \\ & \wedge \text{EntitySkICats}(\text{container}) = \text{SessionSkICats}(\text{session}) \\ & \wedge (\exists S \cdot S \subseteq \text{dom}(\text{ContainerContent}) \wedge \{y \mapsto x \mid x \in \text{dom}(\text{ContainerContent}) \wedge \\ & y \in \text{ran}(\text{ContainerContent}(x))\} [S] = \text{Su}\{\text{folder}\} \\ & \wedge (\forall o \cdot o \in S \Rightarrow ((\text{SessionSkILevel}(\text{session}) \geq \text{EntitySkILevel}(o) \\ & \wedge \text{EntitySkICats}(o) \subseteq \text{SessionSkICats}(\text{session})) \\ & \vee \text{QSR}(o) = \text{FALSE})) \\ & \wedge ((\text{SessionIntegrity}(\text{session}) \geq \text{EntityIntegrity}(o)) \\ & \vee \text{QNR}(o) = \text{FALSE}) \\ & \wedge (\exists r \cdot r \in \text{CurrentRoles} \wedge r \mapsto \text{ReadA} \in \text{SessionRoleAccesses}(\text{session}) \wedge \\ & o \mapsto \text{Execute} \in \text{RoleRights}(r)))) \end{aligned}$$

where most of these identifiers are variables and event parameters.

Complicated predicates

A **solution** that can be used right now. Not the best one because it looks awkward and complicates the proof.

```

tempPredicate ∈ Entities → (Sessions → (P(Entities) → ((Entities → (Names ↔ Entities)) →
(Integrity → (SkILevels → (P(SkICategories) → (Integrity → (SkILevels →
(P(SkICategories) → (BOOL → (BOOL → (P(Roles) → ((Roles ↔ Accesses) →
(P(Entities) → ((Entities ↔ AccessRights) → BOOL))))))))))))))

∀ container, session, CC, cc, si, scl, scc, ei, ecl, ecc, qsr, qnr, CR, saas, CE, rrr ·
  CC ⊆ Entities ∧ container ∈ CC ∧ session ∈ Sessions ∧ cc ∈ Entities → (Names ↔ Entities)
  ∧ si ∈ Integrity ∧ scl ∈ SkILevels ∧ scc ⊆ SkICategories ∧ ei ∈ Integrity ∧ ecl ∈ SkILevels ∧ ecc ⊆ SkICategories
  ∧ ccr ∈ BOOL ∧ ccri ∈ BOOL ∧ CR ⊆ Roles ∧ saas ∈ CR ↔ Accesses ∧ CE ⊆ Entities ∧ rrr ∈ CE ↔ AccessRights
  ⇒ (ecl(container) = scl(session)
    ∧ ecc(container) = scc(session)
    ∧ ((∃ S · S ⊆ CC ∧ {y ↦ x | x ∈ CC ∧ y ∈ ran(cc(x))}[S] = SU{container}
      ∧ (∀ o · o ∈ S ⇒ ( (scl ≥ ecl
        ∧ ecc ⊆ scc)
        ∨ qsr = FALSE)
      ∧ ((si ≥ ei)
        ∨ qnr = FALSE)
      ∧ (∃ r · r ∈ CR ∧ r ↦ ReadA ∈ saas ∧ o ↦ Execute ∈ rrr) )))
  ⇔ tempPredicate(container)(session)(CC)(cc)(si)(scl)(scc)(ei)(ecl)(ecc)(qsr)(qnr)(CR)(saas)(CE)
  (rrr) = TRUE)

```

A proper **solution**: something like macros in C language.

Limitations of text editors

Feature	Camille	Rodin editor
Copy/paste	+	-
Manual development without using a mouse	+	±
Syntax highlighting	+	-
Speed	-	+
Stability	-	+
Support Rodin 3.0	-	+

Some difficulties that we faced

Difficulties in developing the model:

- Complicated predicates
- Limitations of text editors

Difficulties in proving the model:

- Impossibility of team verification
- Excessive number of automatically added hypotheses
- Too much time spent on auto proving
- Auto tactics



Impossibility of team verification

Some facts about our model:

- Consists of only two files (one context and one machine)
- Up to 2 days for proving some proof obligations
- More than a thousand proof obligations
- More than 200mb on a single file with proofs

Problem: These reasons make it difficult to use version control systems.

Solution: Split files with proofs into several small files, e.g. one proof obligation per a file.

Excessive number of automatically added hypotheses

Problem: A large number of automatically added hypotheses to the proving perspective greatly complicates proofs.

Solution: To discuss ways to sample required hypotheses more intelligently.

Too much time spent on auto proving

Problem: Auto proving of the entire model can easily take several hours.

Solution: To parallelize this process both at the level of proof obligations and at the level of proof trees.

Auto tactics

For example, let's look at this hypothesis:

```
x1=deleteAsRoles(x0)
```

When I see something like that during proving of our model I know that I must use the following hypothesis, always:

```
 $\forall i, r. i \mapsto r \in \text{deleteAsRoles} \Rightarrow i \mapsto r \in \text{UserAsRoles}(\text{user})$ 
```

Problem: Currently there is no way to automate such steps.

Solution: The capabilities of Rodin auto tactics can be extended by supporting means for writing your own proof tactics, especially for your model. This idea is similar to PVS Proof Strategies.

Summary



Despite all these issues Rodin helped us:

- To develop the model for the system with a large number of dependences between its objects
- To reveal a number of inaccuracies in the initial system description
- To prove correctness of quite a complicated model for this system

Questions


Now we are going to develop and prove a model for another system. It would be great if we can avoid difficulties that we faced during our past work:

- Complicated predicates
- Limitations of text editors
- Impossibility of team verification
- Excessive number of automatically added hypotheses
- Too much time spent on the auto proving
- Auto tactics

What do you think about this?



Thank you!

 Alexey Khoroshilov,
Ilya Shchepetkov
{khoroshilov,shchepetkov}@ispras.ru

ISPRAS

Institute for System Programming of the Russian Academy of Sciences