Partial Models: A Position Paper

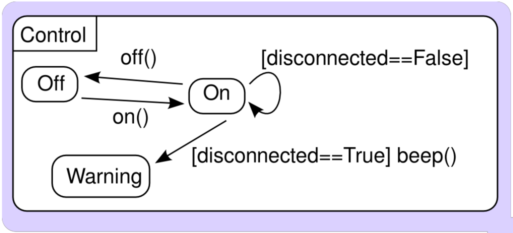
Michalis Famelis, Shoham Ben-David, Marsha Chechik and Rick Salay

University of Toronto

October 17, 2011

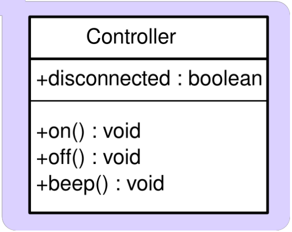
Motivating Example

Bob and Alice are building a network controller:



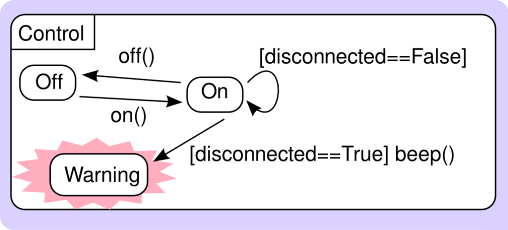
Bob's behavioral model

Alice's architectural model



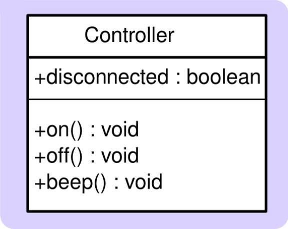
Motivating Example

Bob and Alice are building a network controller:



Bob's behavioral model

Alice's architectural model

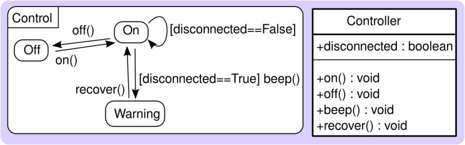


Constraint (C1):

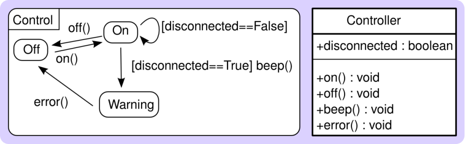
"No sink states."

Bob's Alternative Fixes

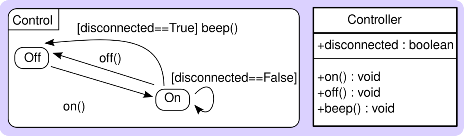
1. Recover back to `On`:



2. Log an error and turn `Off`:



3. Get rid of `Warning`:



Uncertainty: Which Alternative?

Bob has a problem:

- Requirements are unclear about recovery.
- Any changes to the architectural model must be approved by Alice.

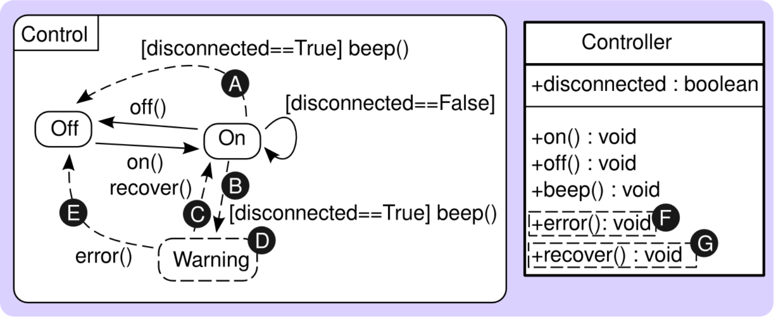
What are his options?

- Stop and wait for more information.
- Make an (informed) guess, risk backtracking.
- Work with the entire set of alternatives.
- Use Partial Models! :)

Note: Inconsistency fixing is merely an example.

There could be other sources of uncertainty!

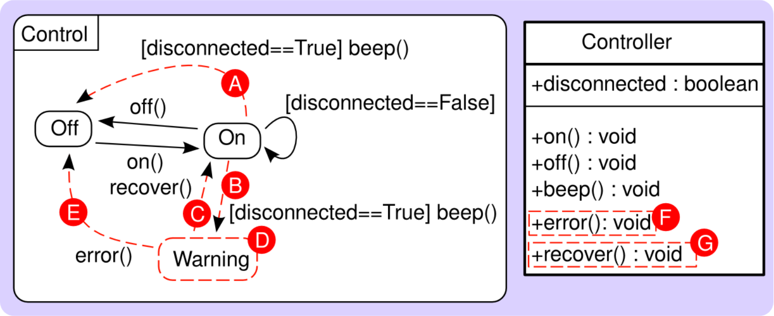
What Is A Partial Model?



$$\Phi p = (\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G) \vee$$
$$(\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee$$
$$(A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)$$

What Is A Partial Model?

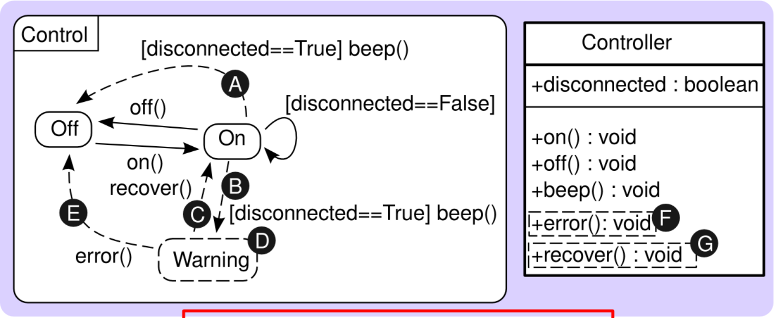
Optional elements



$$\Phi p = (\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G) \vee$$
$$(\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee$$
$$(A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)$$

What Is A Partial Model?

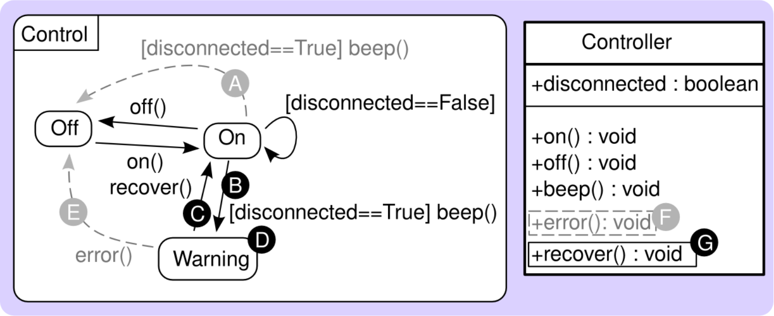
Allowable configurations



$$
\begin{aligned}
\Phi p = {} & (\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G) \vee \\
& (\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee \\
& (A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)
\end{aligned}
$$

What Is A Partial Model?

Concretization



$$\Phi p = \boxed{(\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G)} \vee$$
$$(\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee$$
$$(A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)$$

The Position

Facilitate decision deferral in the presence of uncertainty

by using Partial Models, that represent sets of alternatives,

as first-class development artifacts.

What do we mean by "first-class development artifact"?

- Checking of properties.
- Transformation and refinement.

Comments On Partial Models

Other important characteristics:

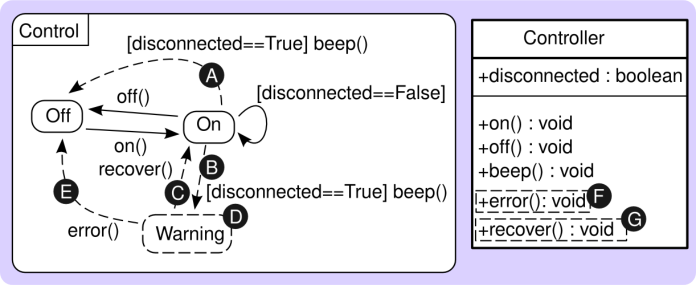
- Compact and exact representation of a set.
- Metamodel/language independence.

Status:

- Different kinds of partiality, submitted [SCF11].
  - May, Abs, Var, OW
- Construction algorithm, submitted [FSC11].
- Preliminary implementation with Alloy/KodKod.

Checking Properties

Check C1 ("no sink states") on the partial model M_1:

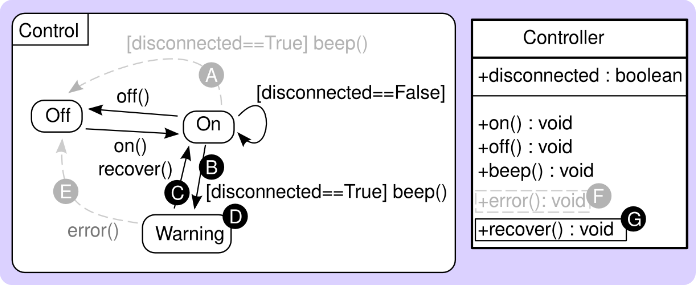


$$\begin{aligned}\Phi p = {} & (\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G) \vee \\ & (\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee \\ & (A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)\end{aligned}$$

It holds for all concretizations. Result: **True**.

Checking Properties

Check C2 ("no transitions with identical source and target"):

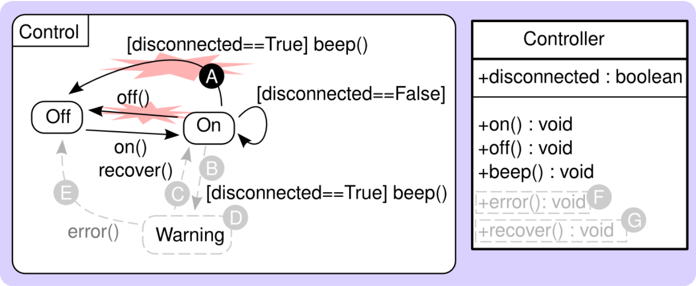


$$\Phi p = \boxed{(\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G)} \vee (\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee (A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)$$

C2 holds for some concretizations.

Checking Properties

But C2 does not hold for others:



$$\Phi p = (\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G) \vee (\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee \boxed{(A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)}$$

The result is therefore **Maybe**.

How Is Checking Done?

- High level algorithm:
  1. Express the *entire* partial model as a formula Φ_{M1}:

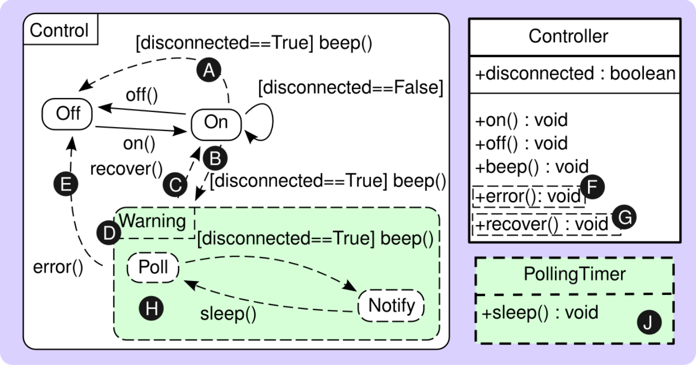
     $\Phi_{M1} = \Phi_P \wedge \texttt{Control} \wedge \texttt{Off} \ldots \wedge \texttt{Controller} \wedge \texttt{on()} \wedge \ldots$
  2. Express the property as a propositional formula Φ_{C2}.
  3. Check $\Phi_{M1} \wedge \Phi_{C2}$ and $\Phi_{M1} \wedge \neg\Phi_{C2}$ for SAT.
- If SAT, we also get counterexamples for feedback.
- We can reason about *all* concretizations together, with two queries to the SAT solver.
- Study of feasibility and scalability, submitted [FSC11].

Two Kinds Of Transformations

1. Classical transformations *adapted* to work for Partial Models.
   - "Detail-adding" (DA) refinements, refactoring
   - Allowing development to continue, even in the presence of uncertainty.
2. Transformations *specific* to Partial Models.
   - "Uncertainty-removing" (UR) refinements.
   - A systematic way to incorporate new information.

Using Adapted Transformations

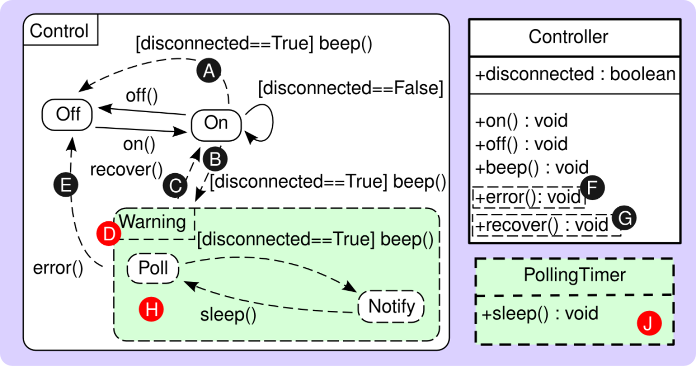
Bob elaborates `Warning` by DA refinement.



$$\Phi p' = \big((\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G) \vee (\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee (A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G) \big) \wedge D \Rightarrow (H \wedge J)$$

Why "Adapt" Transformations?

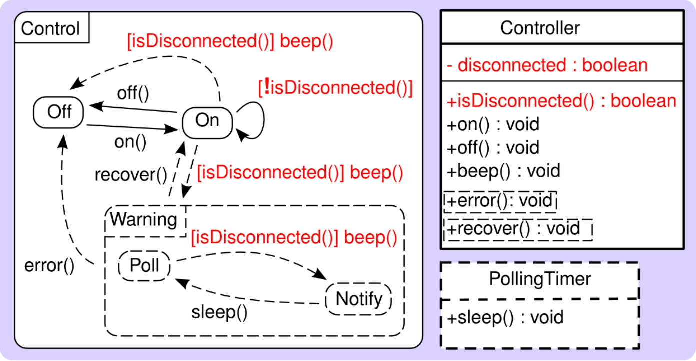
Not all concretizations have a `Warning` state!



$$\Phi p' = ((\neg A \wedge B \wedge C \wedge D \wedge \neg E \wedge \neg F \wedge G) \vee (\neg A \wedge B \wedge \neg C \wedge D \wedge E \wedge F \wedge \neg G) \vee (A \wedge \neg B \wedge \neg C \wedge \neg D \wedge \neg E \wedge \neg F \wedge \neg G)) \boxed{\wedge D \Rightarrow (H \wedge J)}$$

Some “Just Work”

Bob can do some kinds of refactoring straightforwardly:

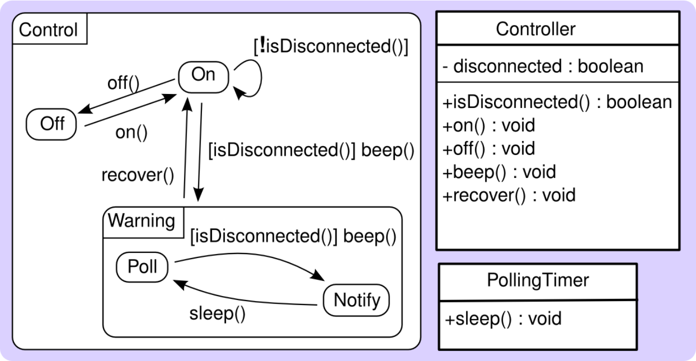


Comments On Adapted Transformations

- Detail-adding refinements should result in models with "more information", same level of uncertainty.
- Refactoring should not add or remove information and/or uncertainty.
- Adapted versions of classical transformations must be total and surjective.
- Such transformations preserve **True** existential and **False** universal properties.

Removing Uncertainty

Once Bob and Alice have negotiated, they can return to classical models:



"Uncertainty-removing" refinements:
Transformations specific to Partial Models.

Comments On UR Refinements

- For May partiality, optional elements can be kept optional, removed or made mandatory.
- Completely remove uncertainty: *Concretization*
- UR refinement: a generic refinement mechanism, with well understood properties [SCF11]:
  - **True** (**False**) properties remain **True** (**False**).
  - **Maybe** properties can be changed into **True** or **False** or remain unaffected.

Summary

- Goal: Facilitate decision deferral in the presence of uncertainty.
- Approach: Use Partial Models to represent sets of alternatives.
- How: Partial Models are first-class development artifacts.
  - Property checking.
  - Adapted transformations.
  - Partial Model-specific transformations.

Conclusion

- Important contributions:
  - Decision deferral in the presence of uncertainty.
  - Compact and exact representation of a set.
  - Metamodel independence.
- In the paper: 14 specific Research Questions.
- Preliminary work on Representation, Property Checking.
- Some prototype tooling.
- Main focus now: Transformations.

Bibliography

Michalis Famelis, Rick Salay, and Marsha Chechik.
Partial models: Towards modeling and reasoning with uncertainty, 2011.

Rick Salay, Marsha Chechik, and Michalis Famelis.
Language independent refinement using partial modeling, 2011.

Available at `http://www.cs.toronto.edu/~famelis`

Questions?