NUI MAYNOOTH
Ollscoil na hÉireann Má Nuad

IRISH RESEARCH COUNCIL
An Chomhairle um Thaighde in Éirinn

A LOGICAL FRAMEWORK FOR INTEGRATING SOFTWARE MODELS VIA REFINEMENT

Marie Farrell

Supervisors: Dr. Rosemary Monahan & Dr. James Power

BACKGROUND

- Formal software engineering is a set of mathematically grounded techniques for the specification, development and verification of software and hardware systems.
- A formal specification is the exact definition in mathematical notation of what the system is required to do (and not do).

EVENT B

- The Event B formal specification language is used in the verification of safety critical systems

- Event B models are an instance of the specification

Machine	Context
variables	carrier sets
invariants	constants
events	axioms

PROBLEM

- Different formalisms do not integrate well e.g. Event B models the specification it does nothing for the implementation and its proofs are not easily transferable to other formalisms

NUI MAYNOOTH
Ollscoil na hÉireann Má Nuad

IRISH RESEARCH COUNCIL
An Chomhairle um Thaighde in Éirinn

SOLUTION

- Establish a theoretical framework within which refinement steps, and their associated proof obligations, can be shared between different formalisms
- Hypothesis: the theory of institutions can provide this framework and, we will construct an institution based specification of the Event B formalism

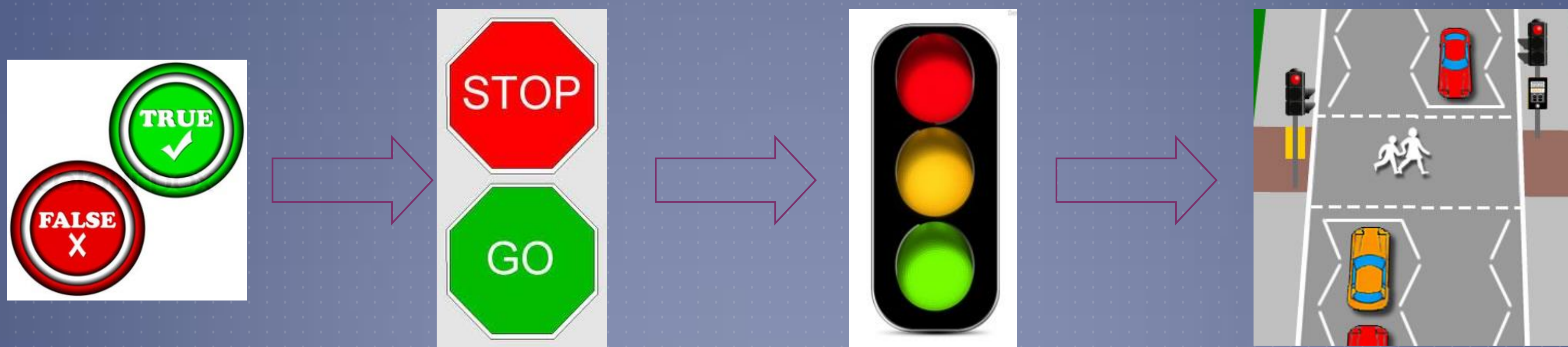
RESEARCH QUESTIONS

1. Can the theory of institutions ensure the accuracy of the translation between Event-B and other specification formalisms?

2. Can this theory allow us to investigate proof obligations generated by Event-B in different formalisms?

REFINEMENT

- Refinement provides a way for us to model software at different levels of abstraction



SOCIAL NETWORK

```
MACHINE
    mac1
SEES
    ctx1
VARIABLES
    person
    rawcontent
    content
    owner
INVARIANTS
    inv1 : person ⊆ PERSON
    inv2 : rawcontent ⊆ RAWCONTENT
    inv3 : content ∈ rawcontent ↔ person
    inv4 : owner ∈ rawcontent → person
EVENTS
    INITIALISATION ≙
        STATUS
      ordinary
BEGIN
      act1 : person := ∅
      act2 : rawcontent := ∅
      act3 : content := ∅
      act4 : owner := ∅
END

    transmit ≙
        STATUS
      ordinary
ANY
      rc
      pe
WHERE
      grd1 : rc ∈ rawcontent
      grd2 : pe ∈ person
      grd3 : rc ↦ pe ∉ content
THEN
      act1 : content := content ∪ {rc ↦ pe}
END

END
```

```
MACHINE
    mac2
REFINES
    mac1
SEES
    ctx1
    ctx2
VARIABLES
    person
    rawcontent
    content
    owner
    visible
    viewpermission
INVARIANTS
    inv1 : visible ∈ rawcontent ↔ person
    inv2 : viewpermission ∈ person ↔ person
EVENTS
    INITIALISATION ≙
      extended
        STATUS
      ordinary
BEGIN
      act1 : person := ∅
      act2 : rawcontent := ∅
      act3 : content := ∅
      act4 : owner := ∅
      act5 : visible := ∅
      act6 : viewpermission := ∅
END

    transmit ≙
        STATUS
      ordinary
REFINES
      transmit
ANY
      rc
      pe
WHERE
      grd1 : rc ∈ rawcontent
      grd2 : pe ∈ person
      grd3 : rc ↦ pe ∉ content
THEN
      act1 : visible := visible ∪ {rc ↦ pe}
      act2 : viewpermission := viewpermission ∪ {owner(rc) ↦ pe}
END
```

REFINEMENT CALCULUS

- Refinement calculus is a notation and a set of rules for deriving programs from their specifications
- Refinement calculii are an extension of Dijkstra's language of guarded commands and both specification and implementation occur within the same formalism
- There are three main theories of refinement:
    1. Carroll Morgan
    2. Ralph-Johan Back
    3. Joseph Morris

MORGAN VS BACK VS MORRIS

- The definition of what constitutes refinement appears to be the same in all calculi
- The rules, however, are slightly different: Morgan is the only one to use miracles
- Back's refinement calculus is much more theoretical that that of Morgan using lattice and category theory as its underlying mathematical basis
- Morris extended Back's refinement calculus to include the notion of prescription
- Since the meaning of what is a valid refinement stays the same then regardless of how it is carried out we should always be able to refine a given specification to an implementation that is semantically consistent across all calculi.

IS THIS REFINEMENT?

- Regular expression → NFA → DFA → min state DFA
- Context free grammar → LR Parser
- Parsing in general
- α- conversion
- β- reduction
- A class extending another class
- A class and an interface it implements
- An interface and another interface it extends
- A generic class/ interface and one of its instantiations
- A class and an instance of the class
- Liskov Substitution
- Refactoring
- UML with OCL → C#/with contracts
- Is refinement a consequence relation á la Tarski?

GENERAL THEORY OF REFINEMENT

- REEVES AND STREADER 2008

- The general model takes as primative:
  1. A set of entities: the specifications and implementations we wish to develop by refinement
  2. A set of contexts: the environment with which the entities interact
  3. A user formalised by defining the set of observations that can be made when an entity is executed in a given context
- The general definition of refinement is parameterised by a set Ξ of possible contexts and a function O which determines what can be observed
- The concrete entity **C** is a refinement of an abstract entity **A** when no user of **A** could observe if they were given **C** in place of **A**.

DEFINITION

- Let Ξ be a set of contexts each of which entities **C** and **A** can communicate privately with, and O be a function which returns a set of traces, each trace being what a user observes of an execution then:

$$\boldsymbol{A} \sqsubseteq_{\Xi,O} \boldsymbol{C} \triangleq \forall x \in \Xi . O([\boldsymbol{C}]_x) \subseteq O([\boldsymbol{A}]_x)$$

- Since general refinement has contexts Ξ as a parameter, by changing Ξ we are able to model different types of interaction
- This definition of refinement can be further specialised for refinement of specific cases

VERTICAL REFINEMENT

- We can view each special model of refinement as a layer in the grand scheme of things each encompassing a set of entities and a refinement relation
- Mathematically our vertical refinement is a Galois connection between the layers.
- This allows us to interpret high level entities as low level entities using a semantic mapping, however, these low level entities cannot interact with the high level ones so the contexts must also be refined

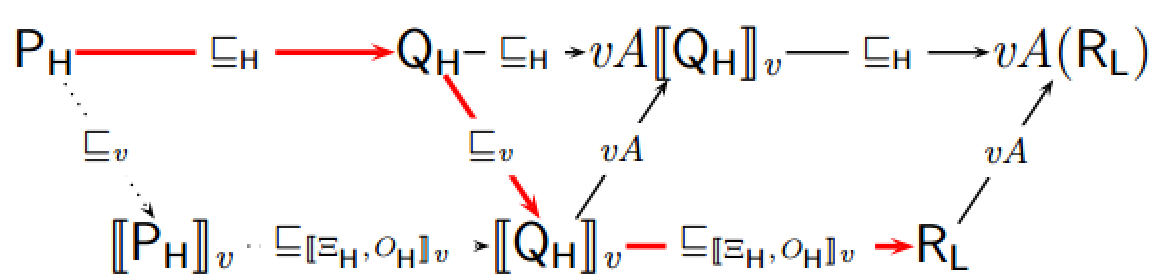
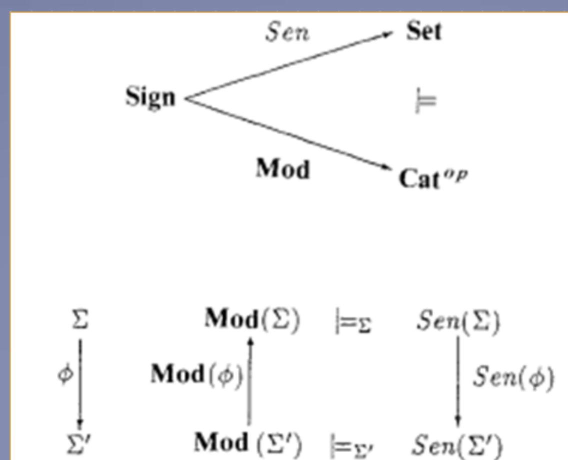


Fig Refinement $\longrightarrow$ within and between layers

CATEGORY THEORY / INSTITUTIONS

- Category Theory is a special branch of Mathematics that allows us not only to describe objects but also to investigate the relationships between them
- Institutions are an application of category theory that allow us to relate the syntactic and semantic structures of different formal languages



Π - INSTITUTIONS

- Alternative to institution – replacing the notions of model and satisfaction by Tarski's consequence operator

- Definition:
  - A π-institution is a triple (Sign, φ, $\{Cn_{\Sigma}\}_{\Sigma:Sign}$) consisting of
    1. A category Sign (of signatures)
    2. A functor φ:Sign -> Set (set of formulae over each signaure)
    3. For each object Σ of Sign, a consequence operator Cn_{Σ} defined in the power set of φ(Σ) satisfying for each A, B ⊆ φ(Σ) and μ: Σ -> Σ

(RQ1) $A \subseteq Cn_{\Sigma}(A)$	(Extensiveness)
(RQ2) $Cn_{\Sigma}(Cn_{\Sigma}(A)) = Cn_{\Sigma}(A)$	(Idempotence)
(RQ3) $Cn_{\Sigma}(A) = \bigcup_{B \subseteq A, B\ finite} Cn_{\Sigma}(B)$	(Compactness)
(RQ4) $\varphi(\mu)(Cn_{\Sigma}(A)) \subseteq Cn_{\Sigma'}(\varphi(\mu)(A))$	(Structurality)

NUI MAYNOOTH
Ollscoil na hÉireann Má Nuad

IRISH RESEARCH COUNCIL
An Chomhairle um Thaighde in Éirinn

CONCLUSION

- Work to date:
  - Denotational Semantics
  - Communicating Sequential Processes (Hoare)
  - Category Theory/Institutions/π-institutions
  - Refinement: Morgan, Back, Morris, general refinement
  - Questions: Is this refienment?
  - Tarski Consequence and refinement
- Work for next semester:
  - Reading course and project in category theory
  - Developing an institution for Event B
  - Developing Event B case studies


ANY
QUESTIONS
?