MT^2 Classification of Discussion Points

- Can we sensibly appropriate concepts of mutation testing in the context of transformations?
  - Why not translate transformations to Java and mutate the Java?
  - What do we need to apply mutation testing to a new domain? Test suites and oracles, execution, mutation operators
  - What should we be mutating in which scenario? Input models, output models, transformations, specifications.
  - Is model transformation a small beautiful and really important application for mutation testing? Is there a killer application for model transformations (not UML2Java), and does it make sense to mutation test it?

- Identifying mutation operators
  - Are there language specific fault taxonomy?
    - Established fault taxonomies for model transformations? Examples – lots of errors in OCL.?
  - Transformation langue flaws versus transformation language implementations.
  - How to choose mutation operators?
  - Assessing utility of different operators sets?
  - Systematic mutation derivation for transformation notations? Hazops.

MT^2 Classification of Discussion Points

- Testing model transformation
  - Oracles?
    - Sloppy/loose oracles?????
    - Possibility of round tripping A->B->C->A via various transformations and also back to back testing.
    - Given two models can you predict the differences between transformed models and their differences in their behaviours?
    - Uses of metamorphic testing ideas and metamorphic relations?
  - [Static analysis + mutation testing to check characteristics]

- Testing models themselves
  - [Mutation of dynamic artefacts is common, but how about mutation of static artefacts?]

- Coverage metrics for testing model transformations
  - Coverage of input metamodel/output metamodel (cause effect graphing)?
  - Transformation language coverage metrics? Shouldn't these exist by now?????
  - Coverage of orderings of rules. (a la method sequences)

- Transformation Language Engineering
  - Lingua Franca (assembly language) for Trans Languages.
  - OCL as an intermediate languages.
  - There has been some work done on formally defined transformations. Especially for graph transformations
  - How well defined are the semantics of transformation languages?
  - Variations in underpinning semantics. Mutation testing to target these things?

THE UNIVERSITY of York

MT^2 Classification of Discussion Points

- Theoretical versus practical interests
  - Scalability

- Applying model transformation for testing
  - Transformations for specific test goals.
  - Model transformations for tools creation.
  - Can we help with tools generation when more sophisticated mutation generation? Knowledge of type systems etc.

Unclassified items

- Issues of execution environments?
- New modelling languages to mutate?
- Feasibility and performance?
  - Purposes of mutation. Abstraction and then mutate?
  - Can you transform between deterministic models to stochastic models usefully (and how does this fit with mutation), How complete are rule sets.
  - Particularisations/specialisations.
- What was meant by?
  - Invariant synthesis??? For transformed constraints.
  - Invariant synthesis and suitability checking.