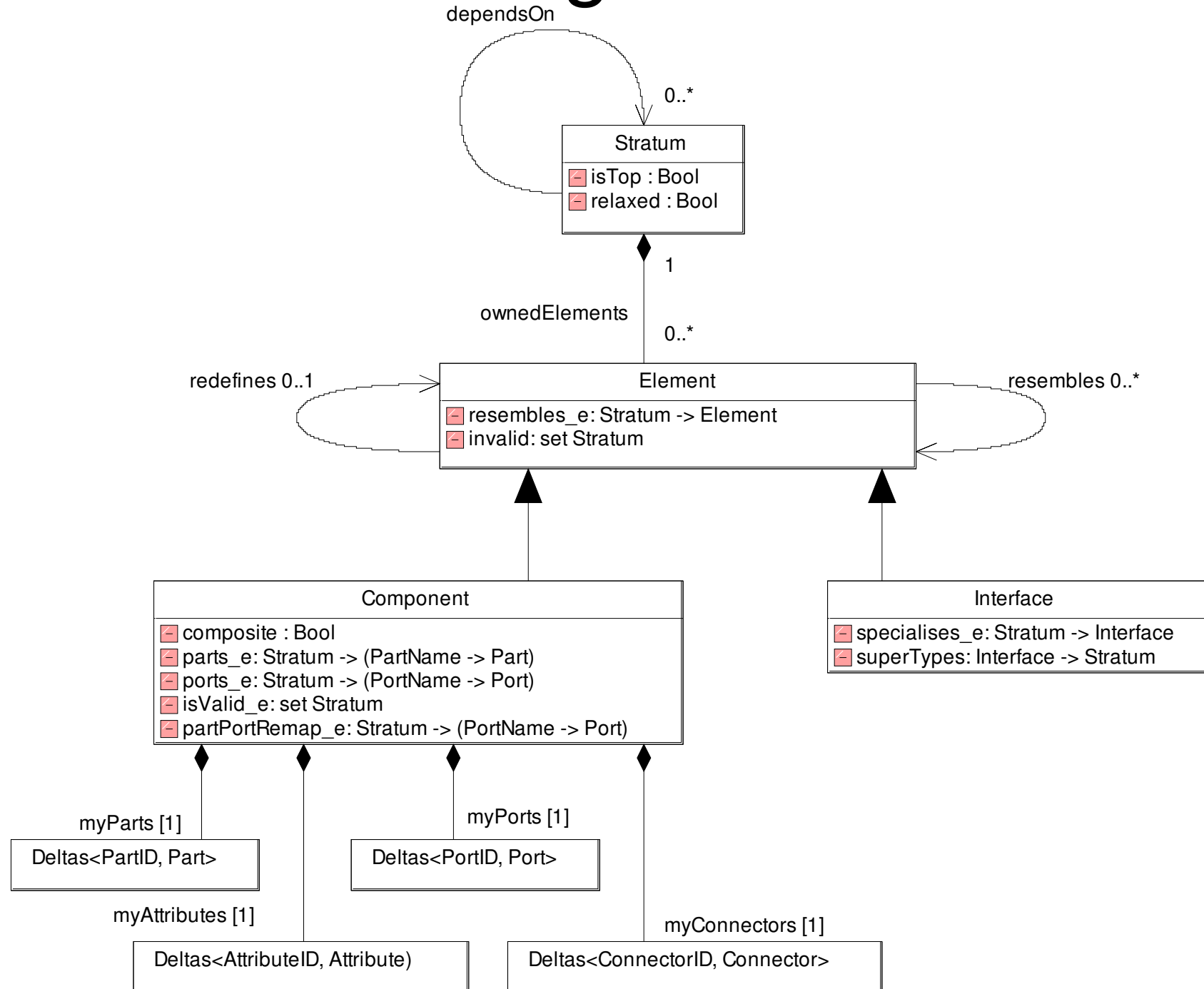


Modeling Backbone Resemblance and Redefinition in Alloy

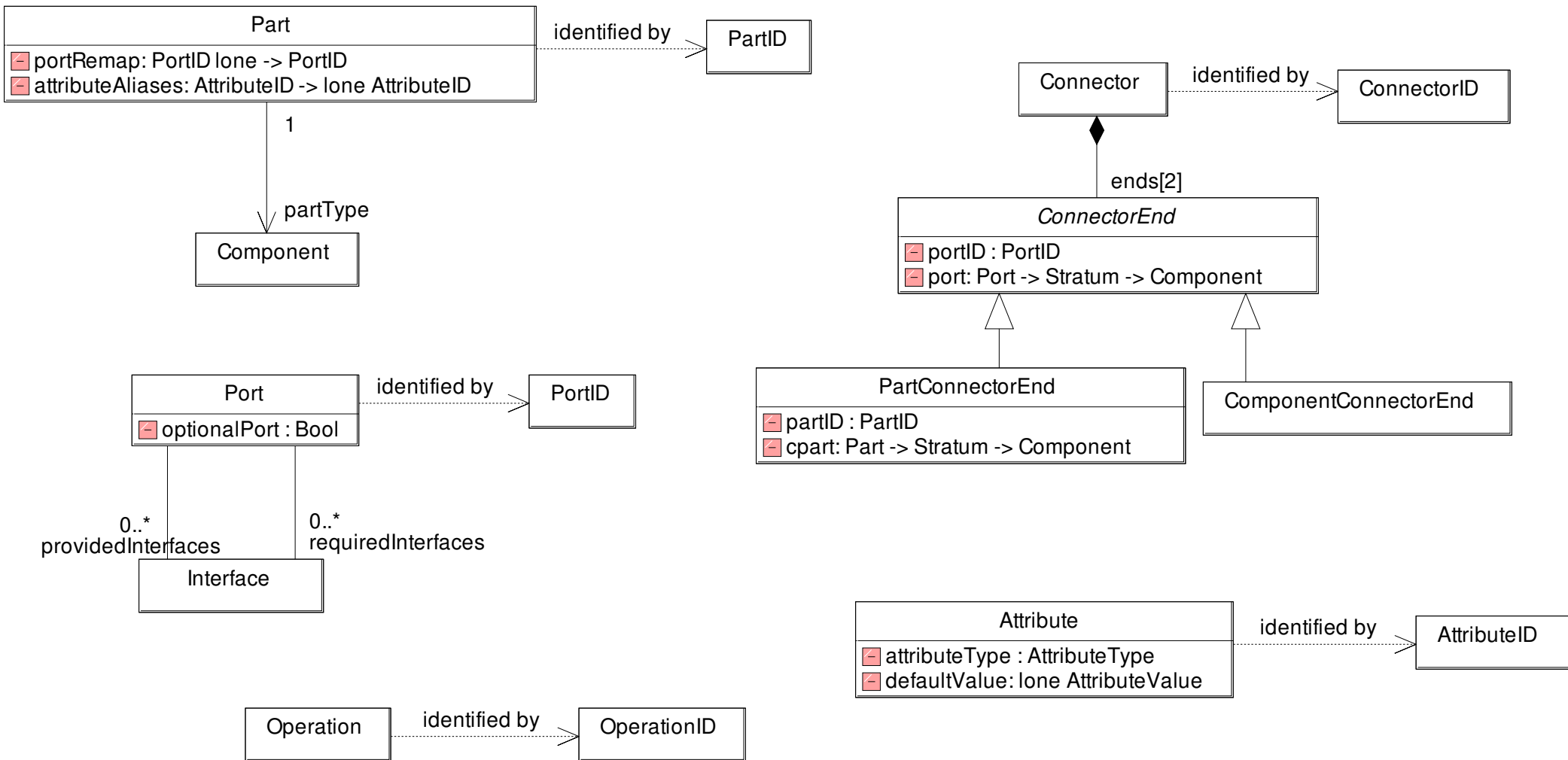
Andrew McVeigh, 24th April 2007

Structure of the model

















Main signatures



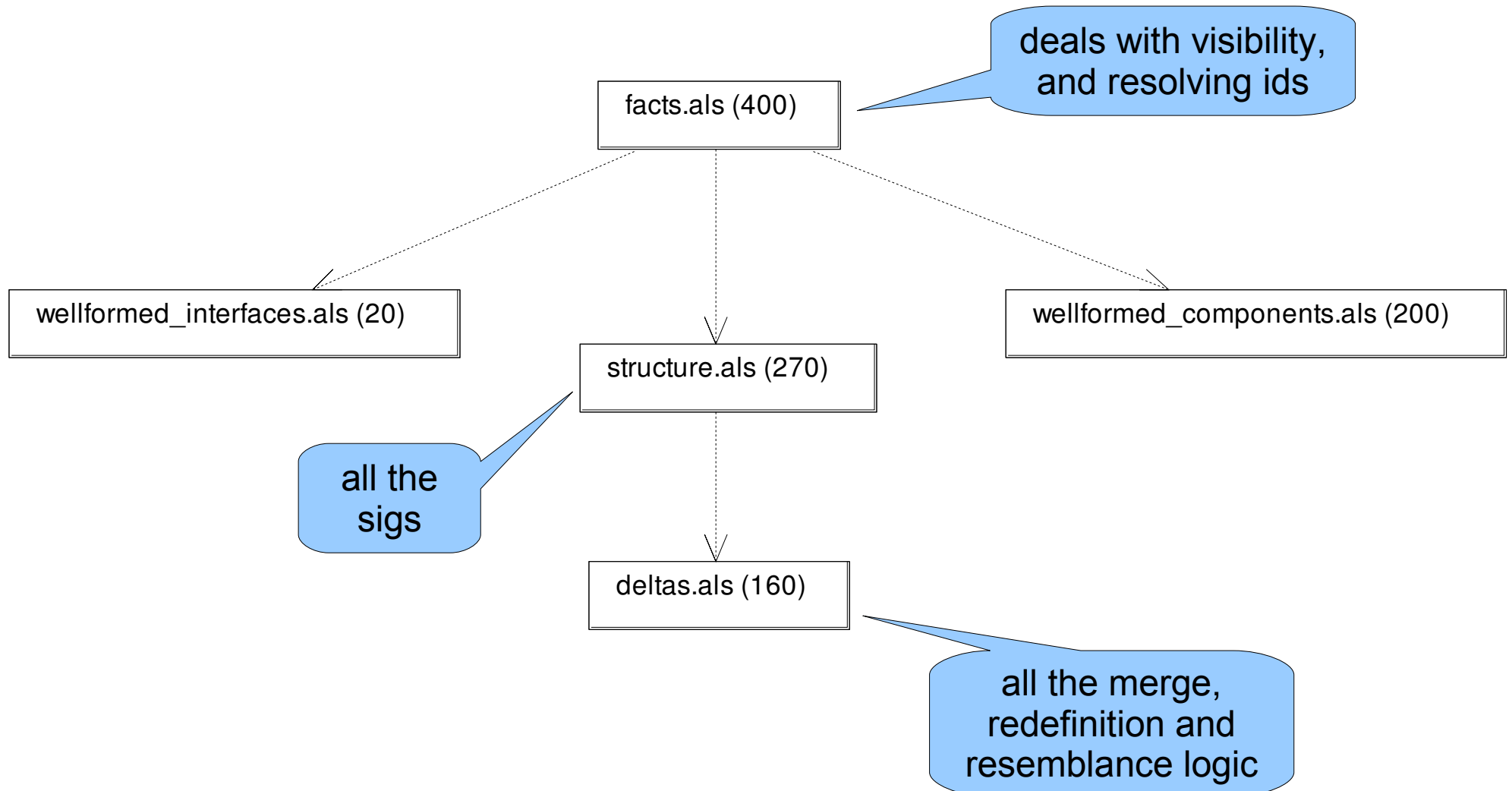
Contained elements...



1 parameterised module for modeling deltas and merging...

Deltas<Stratum, Element, ID, Object>	
	addObjects: newIDs one -> one addedObjects
	replaceObjects: ID one -> lone replacedObjects
	deleteObjects: set ID
	newObjects: set Object
	addedObjects: set newObjects
	replacedObjects: set Object
	newIDs: set ID
	objects_e: Stratum -> ID -> Object
	replacedObjects_e: Stratum -> ID -> Object
	deletedObjects_e: Stratum -> ID
	originalObjects_e: Stratum -> ID -> Object
	originalReplacedObjects_e: Stratum -> ID -> Object
	originalDeletedObjects_e: Stratum -> ID
	predicate deltasWellFormed()
	pred mergeAndApplyChangesForResemblance()
	pred mergeAndApplyChangesForRedefinition()

Module structure of Alloy model



Concepts and Findings

How does it work?

- IDs add a needed level of indirection
 - but approach allows us to refer directly to components
- 2 stages for removing resemblance / redef:
 - rewrite resemblance graphs for each stratum to take redefinition into account
 - merge and copy across contained elements so direct references to a component will work
- This allows us now to concentrate on well-formedness...
 - i.e. for each stratum, is a component well formed?

How does merging work?

- Merge works as follows:
- For each component...
 - take the objects from what we directly resemble and add them all into a single relation ID -> Object
 - remove any IDs that any other “resembler” has deleted, unless they have been replaced
 - add, delete and replace our deltas
- After merging, we can ignore resemblance / redefinition
- Finally, check to see if we have 1 Object per ID!

Key findings: Merge and resolution

- A merge error is:
 - not having a single element per ID (i.e. zero or >1)
 - Check is simple

```
pred Deltas::oneObjectPerID(s: Stratum)
{
    let objects = this.objects_e[s] |
        function[objects, dom[objects]]
}
```

objects is a relation
ID -> Object

- Replace or delete can always resolve
 - Delete will remove all objects for an ID
 - Replace will replace all objects for an ID with 1 object

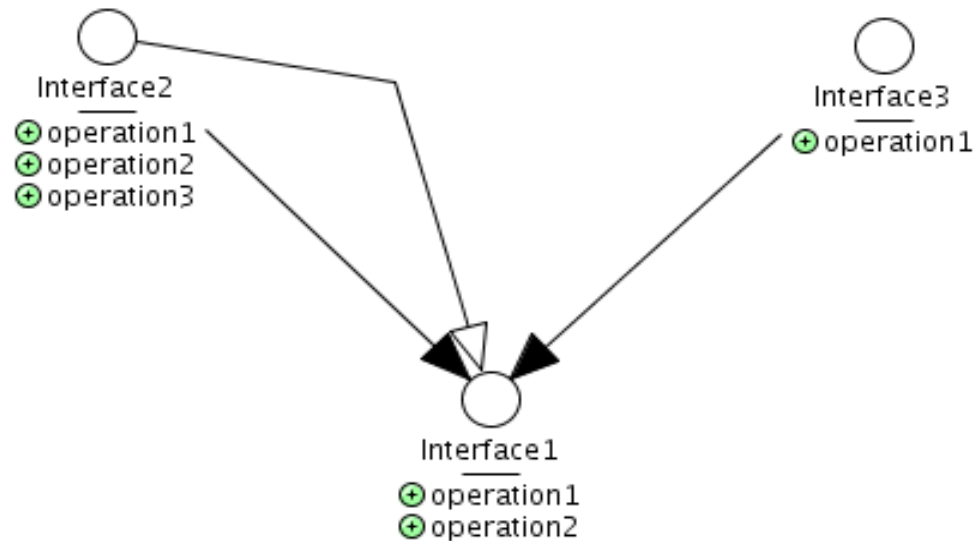
Key findings: Resemblance

- Multiple resemblance is possible
- Redefinition can be rewritten as resemblance
 - Merging is handled via multiple resemblance
- Redefinition and resemblance don't have to align
 - key well-formedness criterion is simple that you don't contain yourself (with a small twist...)

```
let
  original = no c.redefines => c else c.redefines,
  firstResemblesAndComposes =
    c.resembles_e.s - c.redefines + c.parts.s.partType,
  resemblesAndComposes = resembles_e.s + parts.s.partType
{
  original not in firstResemblesAndComposes.*resemblesAndComposes
}
```

Key findings: Interface subtyping

- Resemblance applies to interfaces also
- Subtyping is a subset of resemblance
 - It breaks when an operation is removed or replaced...



```
i.superTypes.s =  
  { super: i.resembles |  
    super.myOperations.objects_e[s] in  
    i.myOperations.objects_e[s] } }
```

Key findings: Modeling approach

- It is not necessary to flatten to model resemblance and inheritance and well-formedness...
 - This is important for the CASE tool
- Interesting finding:
 - Alloy has no recursive functions
 - but, resemblance is best specified as a recursive fn
 - the only way to do this in Alloy is by specifying predicates across all possible parameter combos.
 - leads to an incremental checking approach for the case tool... -> i.e. a +ve from a -ve

Status

- Model for resemblance / redef is complete
 - Can graph in jUMbLe again
 - Can show conflicts and various other properties
- Model is directly applicable to CASE tool
 - about to start refining to implementation...
 - old implementation is complex and incorrect in parts
- Next is flattening semantics
 - Have early Darwin papers to refer to
 - Will rephrase an an Alloy model to refine and implement...
- Also working on transfer report... due next mth.

Other points

- Port remapping
 - required when replacing a part
- Consistent approach to resemblance
 - only components can be resembled / redefined
 - plan to wrap each leaf in a composite
- Global stratum are not possible
 - cannot get rid of cycles...
- SAT solving isn't so bad (maybe...)
 - I can generate snapshots of up to 8 components in < 1hr

Example: Simple redefinition

Delta view

Stratum3

```
redefine-component Component0 (Component2) resembles Component0
{
  replace-parts:
    Component1 PartID0;
    Component1 PartID1 (PortID0->PortID1, PortID1->PortID0);
}
```



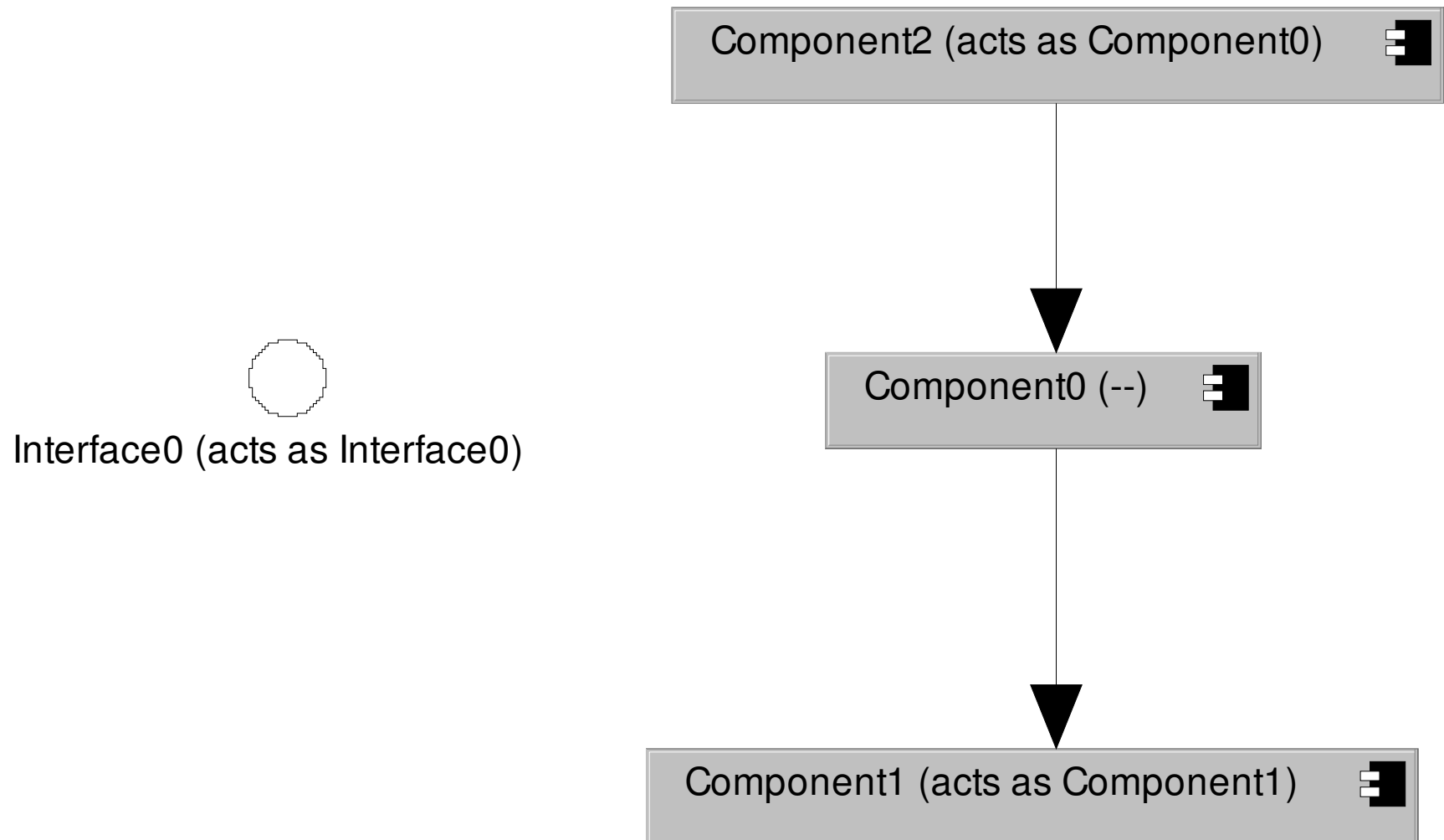
Stratum0

```
interface Interface0
{
  operations:
    OperationID0 Operation0;
  implementation:
    InterfacImplementationID0 InterfacImplementation0;
}
```

```
component composite Component0 resembles Component1
{
  parts:
    Component1 PartID0;
    Component1 PartID1;
  delete-connectors:
    ConnectorID3;
  connectors:
    ConnectorID0 joins PortID1@PartID1 to PortID0;
    ConnectorID1 joins PortID0@PartID0 to PortID1;
    ConnectorID2 joins PortID1@PartID0 to PortID0@PartID1;
}
```

```
component composite Component1
{
  ports:
    PortID0 provides Interface0 requires Interface0;
    PortID1 provides Interface0 requires Interface0;
  connectors:
    ConnectorID3 joins PortID1 to PortID0;
}
```

Resemblance from Stratum3



Final view from Stratum3

