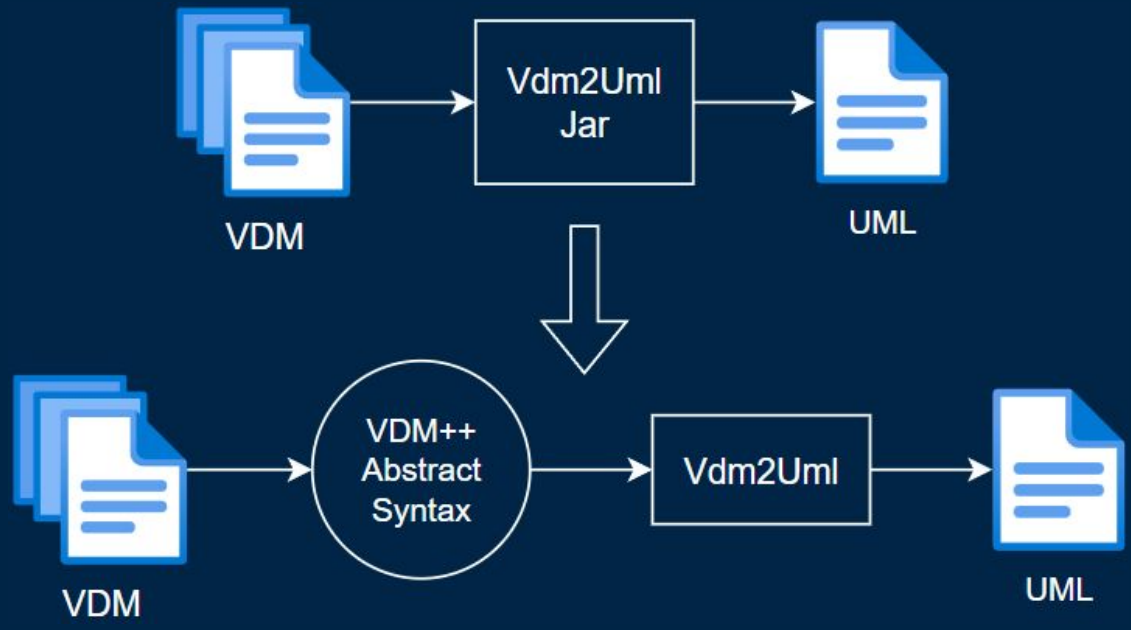
Bidirectional UML Visualisation of VDM Models

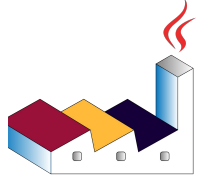
Paper by Jonas Lund, Lucas Bjarke Jensen, Hugo Daniel Macedo and Peter Gorm Larsen

Introduction

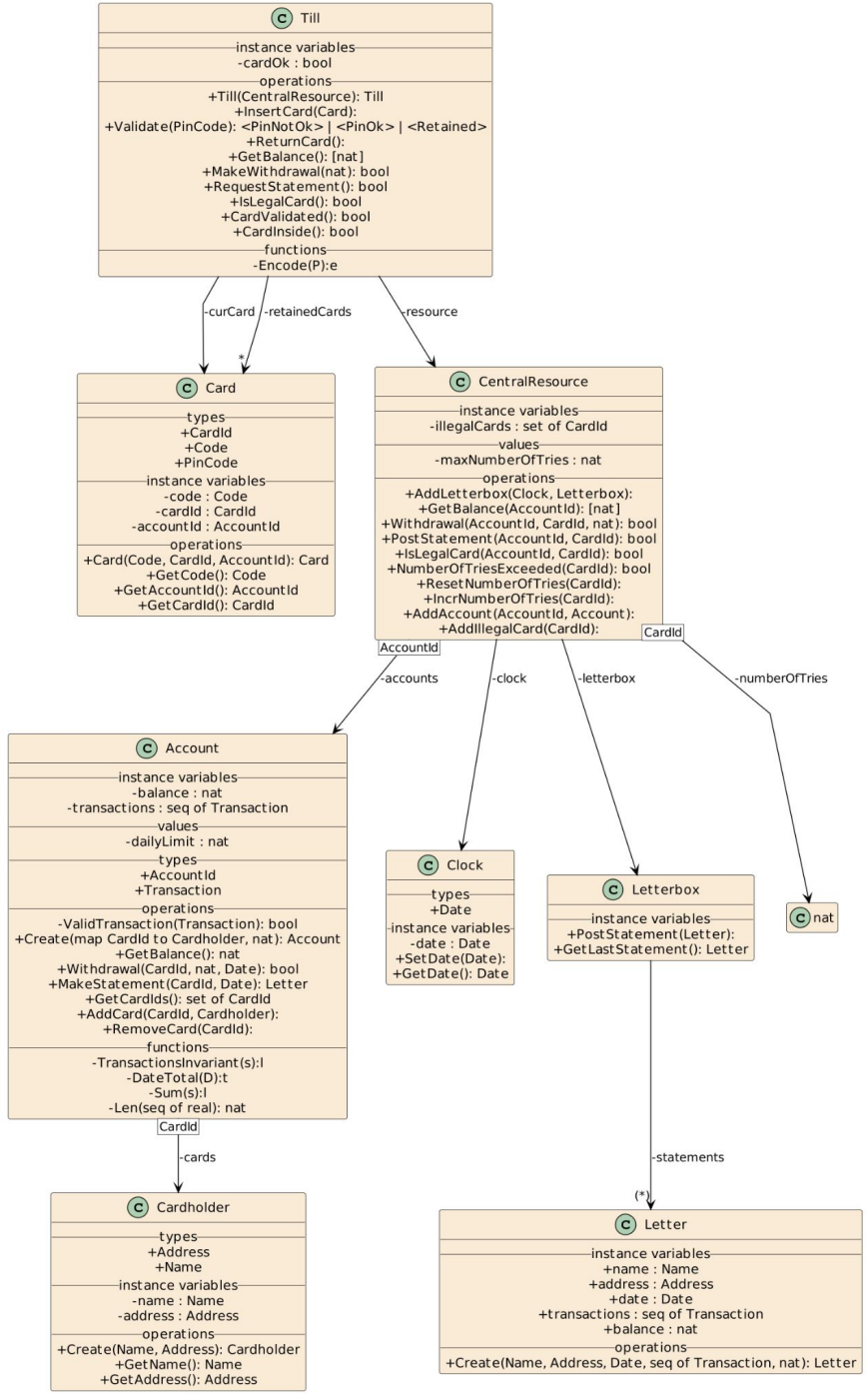


- Progress made at last Overture Workshop
- VDM-PlantUML plugin has been released on VDM VSCode
- Updated mapping rules to describe link between VDM++ and UML Tools
- Defined the subset of PlantUML that describes VDM-models

PlantUML



- Open source textually based diagram tool
- VS Code extension
- Dynamic updates
- Mature and used in the industry



VDM-UML Transformation Rules

- Originally presented in “Connecting UML and VDM++ with Open Tool Support”, Kenneth Lausdahl, Hans Kristian Agerlund Lintrup & Peter Gorm Larsen.
- Now updated and abstracted away from XMI implementation

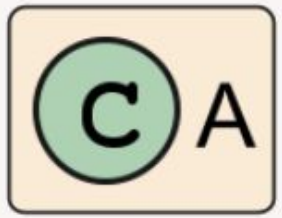
Transformation Rules 1-5: One-to-One Translations

Rule 1: Class Declarations

There is a one-to-one relationship between classes in UML and classes in VDM++.

PlantUML

```
class A{ ... }
```



VDM++

```
class A

...

End A
```

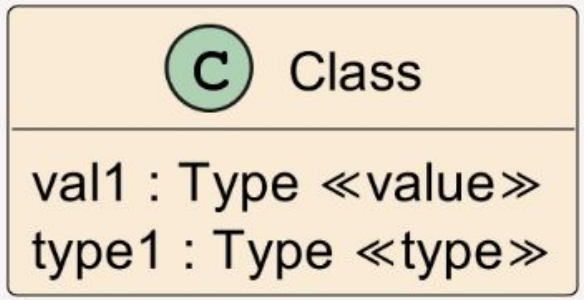
Transformation Rules 2.1 & 3.1 : Stereotypes

2.1: Attribute Stereotypes

Instance variables, types and values are differentiated from each other using stereotypes. If no stereotype is used, the attribute is considered an instance variable.

PlantUML

```
val1 : real «value»
type1 : nat «type»
```



VDM++

```
values
val1 : real = value1

types
type1 = nat
```

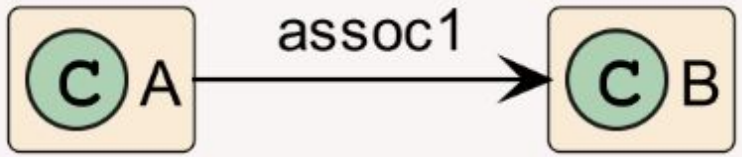
Transformation Rules 6-7 : Class Relations

Rule 7: Associations

Associations between UML classes must be given a role name and a direction. The association is represented in VDM++ as an instance variable in the class at the start of the association, with the role name as its identifier and a value containing a reference to the class at the end of the association.

PlantUML

```
A --> B : assoc1
```



VDM++

```
class A
...
instance variables
assoc1 : B;

end A
```

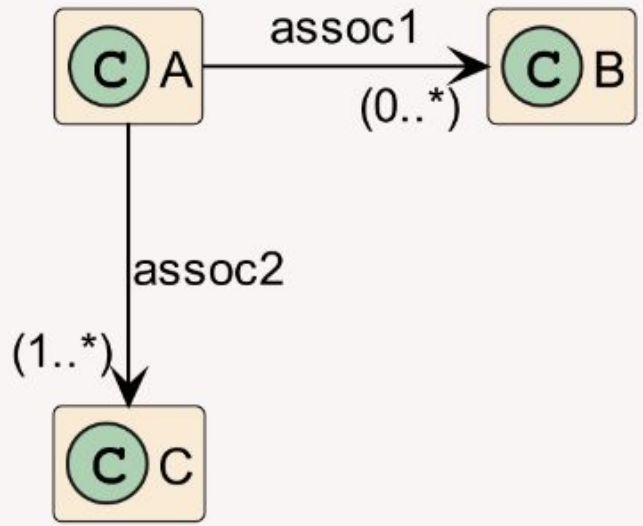
Transformation Rule 8: Association Multiplicity

Rule 8: Association Multiplicity

The multiplicity and ordering of UML relations determine whether the reference to the class at the end of the association is an object reference type or a compound type with an object reference as its sub-type.

PlantUML

```
A --> "(0..*)" B : assoc1
A --> "(1..*)" C : assoc2
```



VDM++

```
Class A

instance variables
assoc1 : seq of B;
assoc2 : seq1 of C;

end A
```

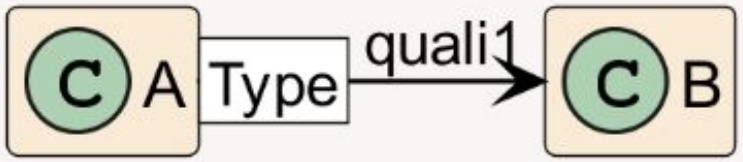
Transformation Rule 9: Qualified Associations

Rule 9: Qualified Associations

Qualified associations use a type (or a class name) as the qualifier and are modelled as an instance variable containing an association from the qualifier type to an end sub-type using the map type.

PlantUML

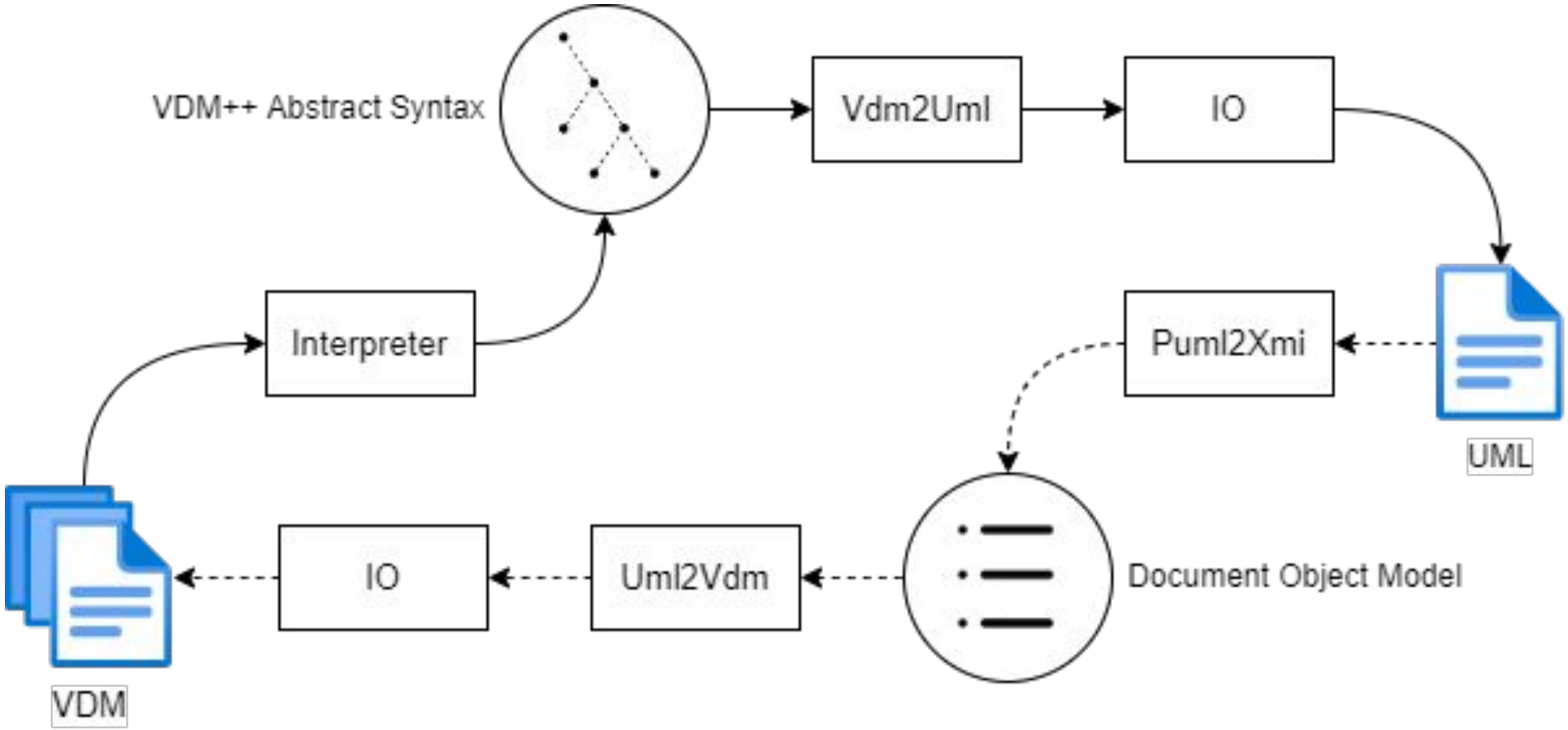
```
A [Type] --> B : quali1
```



VDM++

```
class A

instance variables
quali1 : map Type to B;

end A
```

Implementation

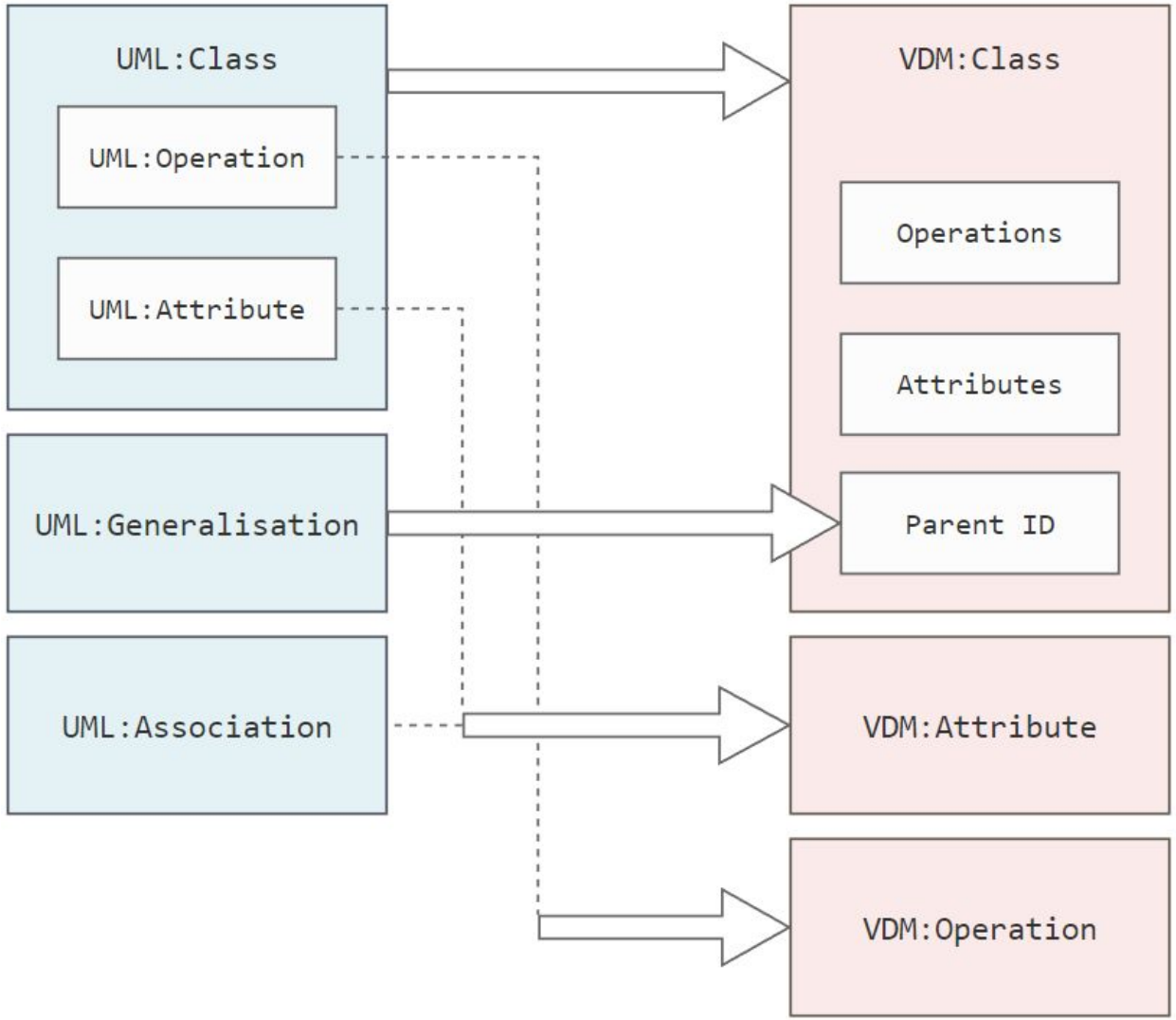


VDM-to-UML

- Implemented as an analysis plugin (can thus use the VDMJ interpreter)
  - Translation Feature on VDM-VSCode
- Traverse the AST using different TC visitors
  - Definition visitor
  - TC Leaf Type visitors
- Compute the cost of types to determine when to do a cutoff
  - Eg. map seq of (char * nat) to set of nat → map seq… to set…

UML-to-VDM

- Convert PUML to XMI
- Parsed using a DOM parser
- Elements extracted and written to VDM file



Demonstration

PlantUML-for-VDM Language Manual

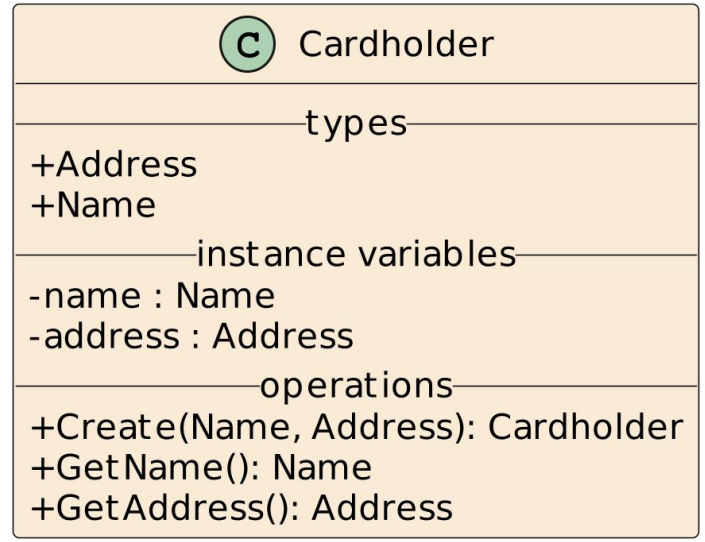
- A Language Manual for the subset of the PlantUML language that correctly describes VDM-models, has been made
- Opens up for static analysis

```
functional definition = operation definition
                      | function definition ;

operation definition = [ access member definition ],
           identifier, '(', [ type ], ')',  ':', type' ;


function definition = [ visibility ], identifier,
        '(', [ type ], ')',  ':', type', '<<function>>' ;
```

Future Work

- VDM-SL and VDM-RT support
- Omit libraries option
- Implicit stereotypes
- Dynamic bidirectionality
  - A means of translating to and from UML without information loss
  - Reacting to model updates bidirectionally
- Static analysis
- Coupling with SysML v2

Future Work



- VDM-SL and VDM-RT support
- Omit libraries option
- Implicit stereotypes
- Dynamic bidirectionality
  - A means of translating to and from UML without information loss
  - Reacting to model updates bidirectionally
- Static analysis
- Coupling with SysML v2

Conclusion

- The plugin is now better integrated with VDM-VSCode
- There is still much work to be done to improve usability

AARHUS
UNIVERSITY