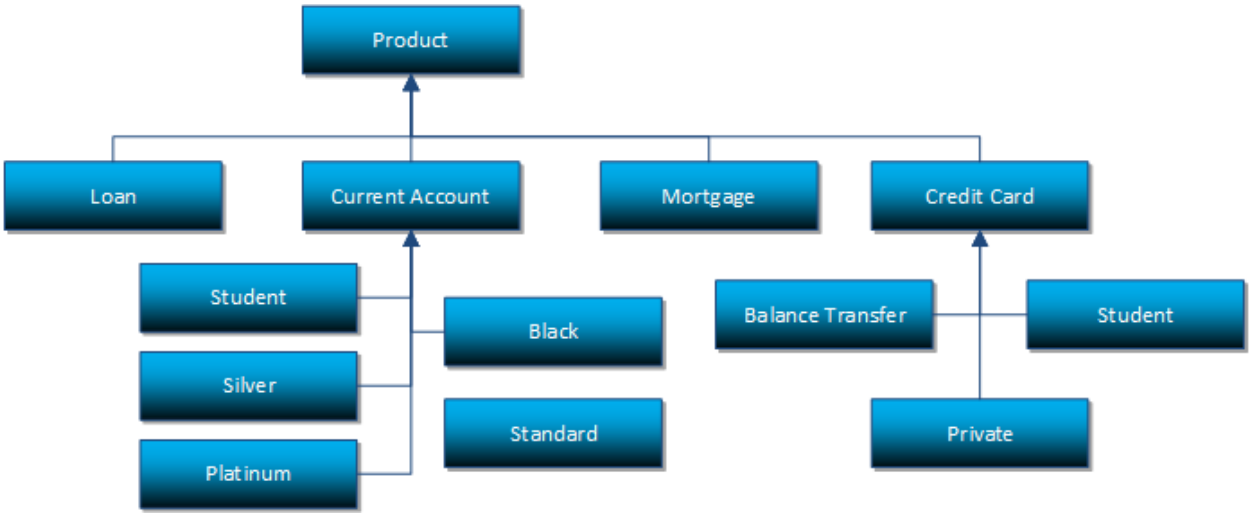
June Challenge

Bruce Silver, methodandstyle.com, using Trisotech and RedHat

Nick's original article came out when DMN was brand new and there weren't any compliant tools. Also, he was trying to illustrate the variety of boxed expression types rather than normal modeling practice. In contrast, my solution is the way I would expect my students today to model, test, and execute Nick's logic.

The scenario logic is very simple. The only tricky part is modeling the Product Holdings input data. Nick offers the following "Fact Model":



I'm not sure if TDM uses something like this but I hope not, as the leaf nodes are a mix of subclasses, child elements, and enumerated values. I assume that the 4 first-level nodes are meant to be subclasses and those below are enumerated values of a hidden child element *Type*. And there is no cardinality indicated; I would assume optional/unbounded. But all this also points out a weakness of DMN's (FEEL) type system, as it has no subclasses or optional elements. So my type *tHoldings* is a sequence of 4 collections, each with various child elements as required by the logic, some of which are possibly null. In reality, each account type would probably be a separate table element, connected in the logic by joins (via account ID), but this complexity was not required by the logic as given. (If you want to see how to do table joins in FEEL, check out DMN Cookbook.)

A couple other stylistic differences from Nick's post:

Like the Chapter 11 example in the DMN spec, Nick goes overboard on BKMs. My view is these should be reserved for when logic is reused or is delegated by the modeler to someone with more technical skill. The BKM to determine if the applicant is an existing customer is used 3 times, but it seems better to just make this a supporting decision and execute it once rather than 3 times. I would expect any

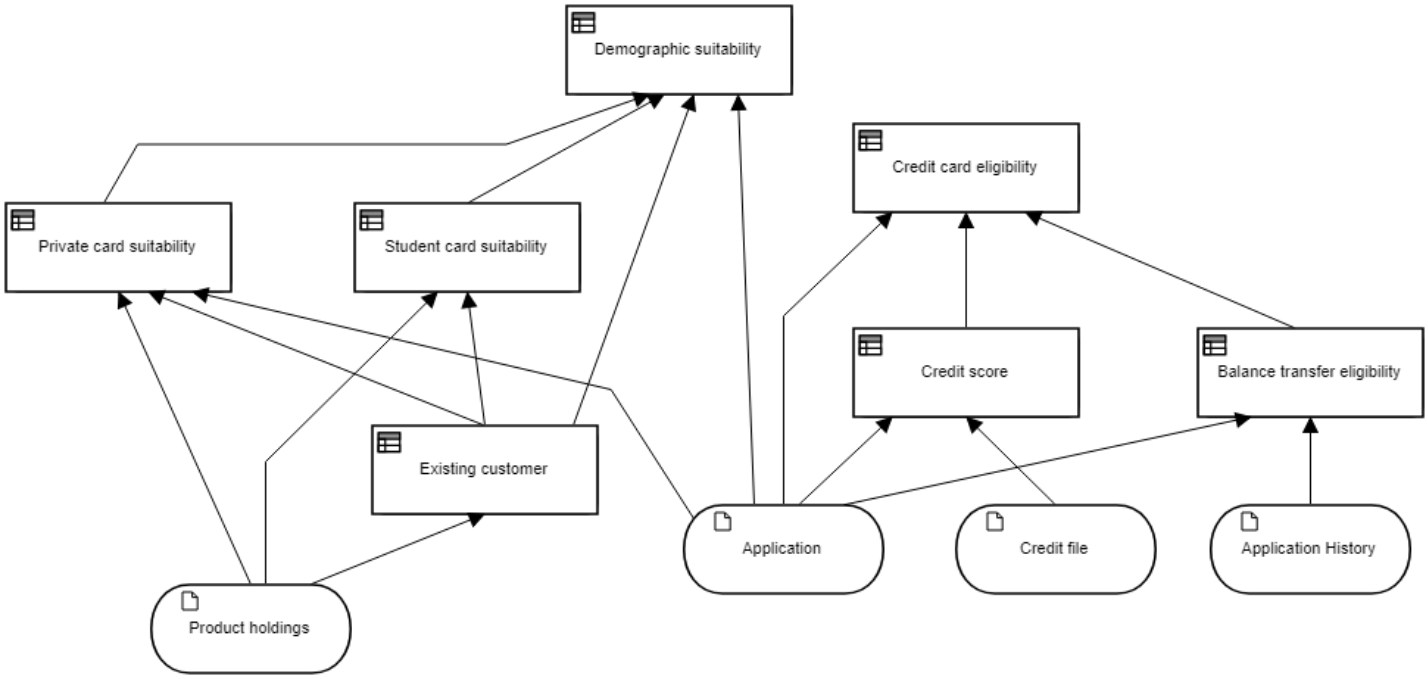
business user with DMN Basics training to be able to handle any of the logic expressions… so no BKMs needed here.

Also, while Nick uses structures for input data, it seems these must be translated into simple types with long stylized names. Is this a TDM thing? It seems like needless busywork. DMN works fine with structured data, just as it is.

Finally, as in Nick's post, the applicant age is calculated from the current date. My solution uses the Trisotech/RedHat extension function today(), which is useful but technically not allowed in FEEL. In reality, the age would be calculated based on an *Application date* field in the input data.

The model xml is provided in 2 ways: a DMN 1.2 compliant form, which includes the graphical layout and test cases per the TCK-standard, and DMN 1.1, without the layout and test cases. It is executable in the tool, and can be deployed in one click as a decision service to Trisotech cloud execution.

DRD



Elements

Demographic suitability (Decision)

Demographic suitability
tSuitability
"Unsuitable", "Suitable"

Applicant age *Number*	(today() - Application.Date of birth)/duration("P365DT6H")

P	Applicant age	Application.Card type	Existing customer	Suitability
	Number	*tCardtype* *"Student", "Private", "Balance transfer"*	*Boolean*	*tSuitability* *"Unsuitable", "Suitable"*
1	<18	-	-	"Unsuitable"
2	-	"Student","Private"	false	"Unsuitable"
3	-	"Student"	true	Student card suitability
4	-	"Private"	true	Private card suitability
5	-	-	-	"Suitable"

Credit card eligibility (Decision)

Credit card eligibility
tEligibility
"Eligible", "Ineligible"

U	Application.Card type	Credit score	Credit card eligibility
	tCardtype	*tCreditscore*	*tEligibility*
	"Student", "Private", "Balance transfer"	*[0..999]*	*"Eligible", "Ineligible"*
1	"Student"	>=500	"Eligible"
2	"Private"	>=750	"Eligible"
3	"Balance transfer"	>=750	Balance transfer eligibility
4	"Student"	<500	"Ineligible"
5	"Private", "Balance transfer"	<750	"Ineligible"

Private card suitability (Decision)

Private card suitability
tSuitability
"Unsuitable", "Suitable"

Total mortgage balance *Number*	if Product holdings.Mortgages != null then sum(Product holdings.Mortgages.Balance) else 0
Total current accounts balance *Number*	if Product holdings.Current accounts != null then sum(Product holdings.Current accounts.Balance) else 0

P	Existing customer	Application.Annual income	Total mortgage balance	Total current accounts balance	Suitability
	Boolean	*Number*	*Number*	*Number*	*tSuitability* *"Unsuitable", "Suitable"*
1	false	-	-	-	"Unsuitable"
2	true	<100000	<300000	<100000	"Unsuitable"
3	-	-	-	-	"Suitable"

```
Suitability
```

Student card suitability (Decision)

Student card suitability
tSuitability
"Unsuitable", "Suitable"

if Existing customer and count(Product holdings.Current accounts[Type="Student"])>0 then "Suitable" else "Unsuitable"

Credit score (Decision)

Credit score
tCreditscore
[0..999]

C+	Credit file.Year Total Defaults	Credit file.Bankruptcy Indicator	Credit file.Years with Bank	Credit file.Credit Used Percentage	Credit score
	Number	*Boolean*	*Number*	*Number*	*tCreditscore [0..999]*
1	0	-	-	-	250
2	(0..3]	-	-	-	100
3	(3..6]	-	-	-	50
4	>6	-	-	-	0
5	-	true	-	-	0
6	-	false	-	-	250
7	-	-	<1	-	50
8	-	-	[1..3]	-	150
9	-	-	>3	-	250
10	-	-	-	[0..25)	200
11	-	-	-	[25..50)	249
12	-	-	-	[50..75)	150
13	-	-	-	[75..100]	100
14	-	-	-	>100	0

Balance transfer eligibility (Decision)

Type	tEligibility *"Eligible", "Ineligible"*

Balance transfer eligibility *tEligibility* *"Eligible", "Ineligible"*		
has previous application last 6 months *Boolean*	Application History.Last application date != null and (today()-Application History.Last application date)/duration("P1D")<180	
Years of address history *Number*	Address1 duration *Days and time duration*	if Application.Address1.EffectiveFrom!=null and Application.Address1.EffectiveTo!=null then Application.Address1.EffectiveTo - Application.Address1.EffectiveFrom else duration("P0D")
	Address2 duration *Days and time duration*	if Application.Address2.EffectiveFrom!=null and Application.Address2.EffectiveTo!=null then Application.Address2.EffectiveTo - Application.Address2.EffectiveFrom else duration("P0D")
	Address3 duration *Days and time duration*	if Application.Address3.EffectiveFrom!=null and Application.Address3.EffectiveTo!=null then Application.Address3.EffectiveTo - Application.Address3.EffectiveFrom else duration("P0D")
	(Address1 duration + Address2 duration + Address3 duration)/duration("P365D")	

P	Application.Annual income	Application.Residential status	has previous application last 6 months	Years of address history	Eligibility table
	Number	*tResidentialstatus* *"UK resident", "Non-UK resident"*	*Boolean*	*Number*	*tEligibility* *"Eligible", "Ineligible"*
1	>10000	"UK resident"	false	>3	"Eligible"
2	-	-	-	-	"Ineligible"

Eligibility table

Existing customer (Decision)

Existing customer
Boolean

count(Product holdings.Credit cards)+count(Product holdings.Current accounts) +count(Product holdings.Loans)+count(Product holdings.Mortgages)>0

Application (Input Data)

Type	tApplication

Credit file (Input Data)

Type	tCreditFile

Application History (Input Data)

Type	tApplicationHistory

Product holdings (Input Data)

Type	tHoldings

Types

tCreditFile

Year Total Defaults	Number
Bankruptcy Indicator	Boolean
Credit Used Percentage	Number
Years with Bank	Number

tCardtype

Text
"Student", "Private", "Balance transfer"

tApplication

Name	Text
Date of birth	Date
Card type	tCardtype
Annual income	Number

Residential status	tResidentialstatus
Address1	tAddress
Address2	tAddress
Address3	tAddress

tCreditscore

Number *[0..999]*

tResidentialstatus

Text *"UK resident", "Non-UK resident"*

tAddress

Street	Text
City	Text
PostalCode	Text
CountryCode	Text
EffectiveFrom	Date
EffectiveTo	Date

tEligibility

Text *"Eligible", "Ineligible"*

tApplicationHistory

Last application date	Date

tCurrentAccount

ID	Text
Type	Text *"Standard", "Student", "Silver", "Platinum", "Black"*
Balance	Number

tLoan

ID	Text
Balance	Number

tCreditCard

ID	Text
Type	tCardtype

Balance	Number

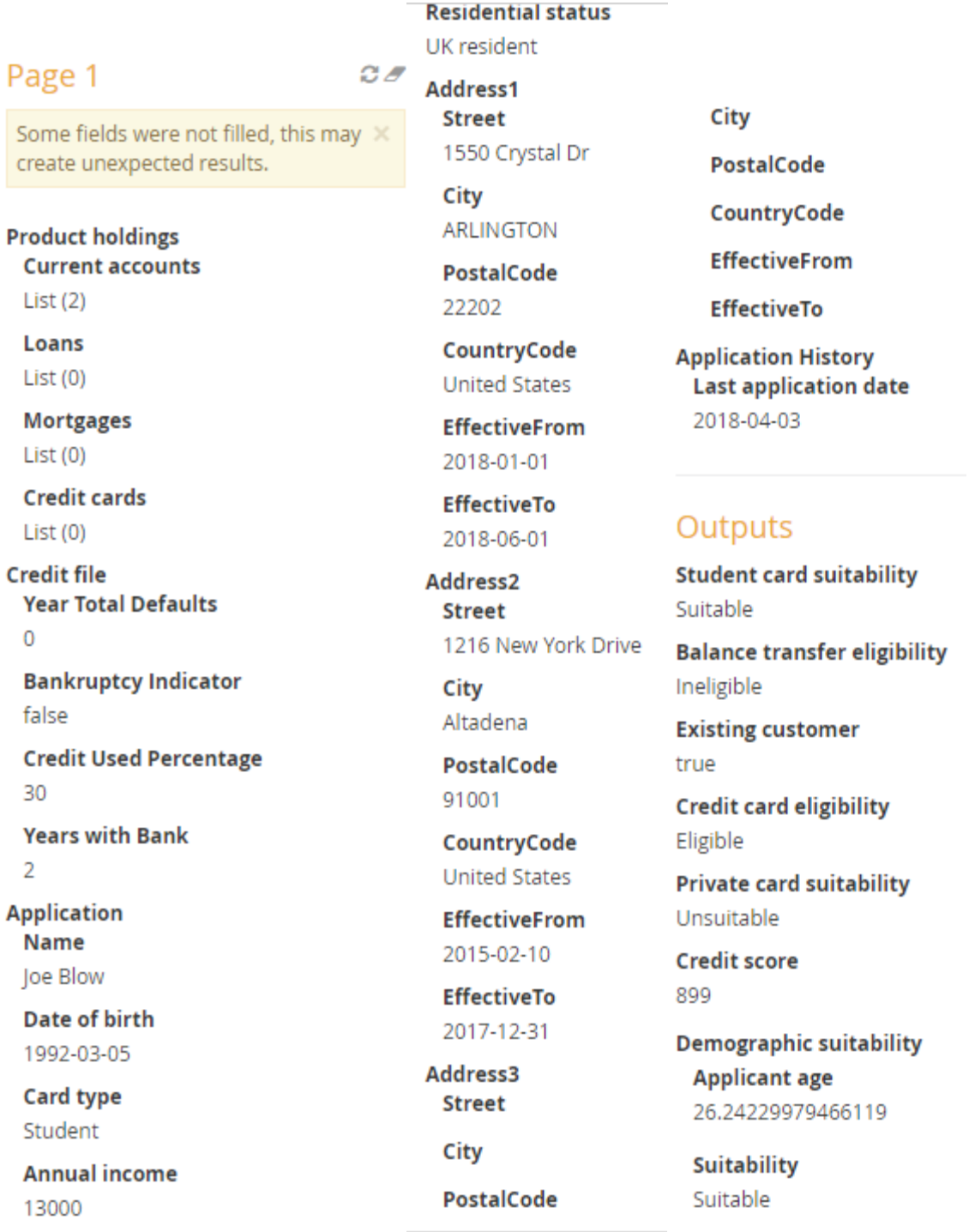
tHoldings

Current accounts	tCurrentAccount
Loans	tLoan
Mortgages	tLoan
Credit cards	tCreditCard

tSuitability

Text *"Unsuitable", "Suitable"*

Test cases



Test Case 2 is identical except Application.CardType=”Private”, with result “Unsuitable”.