High Performance Decision Model Execution

Compilation of DMN into Machine Code

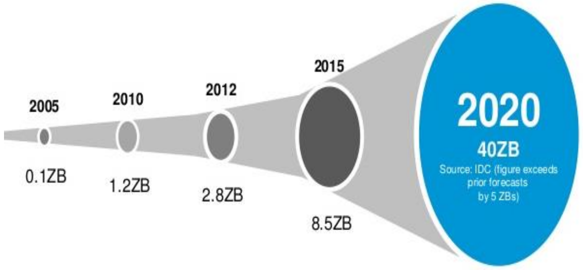
Dr Jolyon Cox and Dr Jan Purchase - www.rapidgen.com - www.luxmagi.com

Overview

- Demand for High-Performance Decision Services
  - Can't be met exclusively with high-spec cloud servers
- We Address
  - What drives the need for high-efficiency decision-making?
  - Which techniques are used to achieve it?
  - What are DMN's barriers to efficiency?
  - How can we overcome them?
- We Demonstrate
  - Anatomy of an example high-performance decision model
  - Compilation and execution of this model
- We'll Take Questions at Any Time

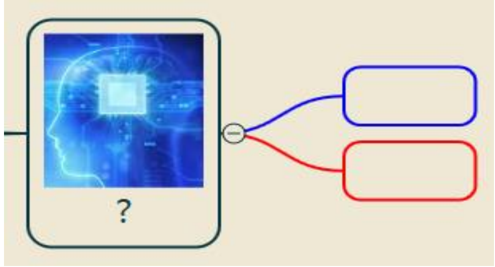
Need For Highly Efficient Decisions

- Driven by
  - Data Explosion: ‘Fine-Grain Decisions’
    - More data sources, personalization



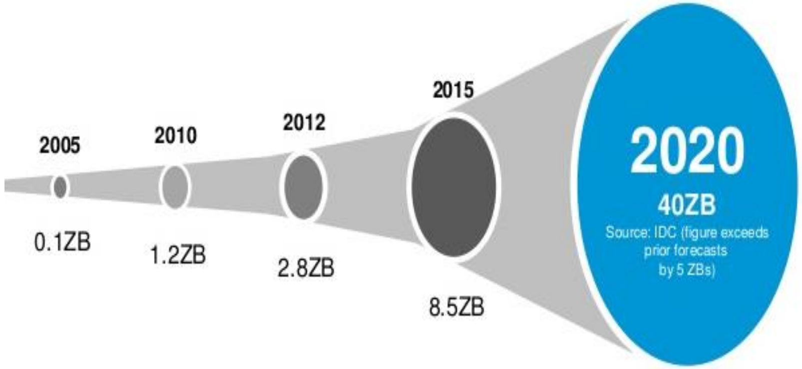
Source: IDC (figure exceeds prior forecasts by 5 ZBs)

  - Complexity of ‘Enlightened decisions’
    - Analytic and AI fuelled decisions



  - Modest hardware available to make ‘edge decisions’
    - Bringing decisions to the data

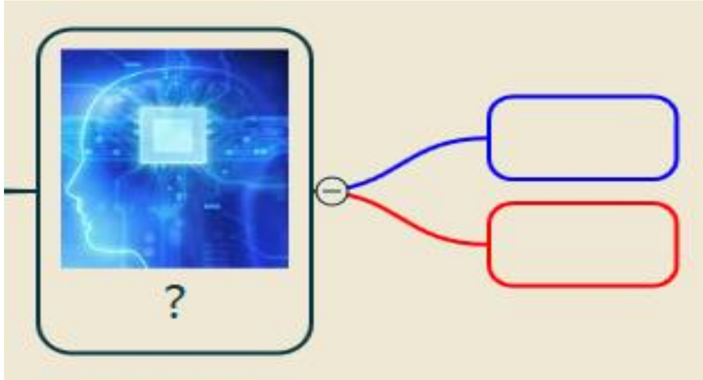
Efficiency Needed to Handle Data Explosion



- Geometrically Increasing Data – 'Fine-Grain' Decisions
  - Data source proliferation
    - Sensors everywhere – IoT
    - Increased access to and application of unstructured data
  - Growing finer-grain applications for decision management
    - Compliance e.g., IFRS-17 contract-by contract processing
    - Personalization e.g., customer journey

Efficiency Needed for ‘Enlightened’ Decisions

- New Data Intensive Applications Driven by AI
  - Inference of customer identity
  - Deduction of customer need and intent
  - Detection of ‘outlier’ behaviour
    - Indicative of future actions, fraud, incipient failure
  - Sensor and input calibration using reinforcement learning
- Need to be ‘Micro-Accountable’
  - For compliance – e.g., GDPR
  - For human interoperability (XAI)



Move Towards Decentralized ‘Edge’ Decisions

- Increased Need to Make Decisions ‘At the Edge’
  - Where data is first manifest and most voluminous
  - Instead of communicating huge volumes to the centre
  - Perform Some Tasks ‘At the Edge’
    - Aggregate, apply analytics, spot local patterns, calibrate
    - Generate knowledge/inferences
    - React, make decisions – subject to centralized confirmation
    - Protect centre from harm (e.g., DoS)
  - Without Real-time Involvement of the Core
- Using 1000s of Low-Spec I/O Devices

Move To Decentralized ‘Edge’ Decisions

- Advantages
  - ‘Natural’ parallelism, distribution of tasks
    - Data collection, cleaning and ethical filtering at source
    - Early aggregation – to minimize transport bandwidth
    - Local sensor collaboration, peer-to-peer ensemble behaviours
  - High redundancy
    - Supports fault detection, self healing networks – no SPoF
    - Less reliance on internet connection
    - Support Byzantine security - resilience to attack
- Applications
  - Remote/hostile environments: in-body, space, deep-sea
  - High-volume/low-signal: surveillance, failure detection

Summing Up…

- Fine-Grain, Enlightened and Edge Decisions
  - Require more efficient decision services
  - Sometimes restricted to modest hardware and connectivity
- An Example Fine-Grain Decision
  - Driven by financial compliance

Example High Performance Decision

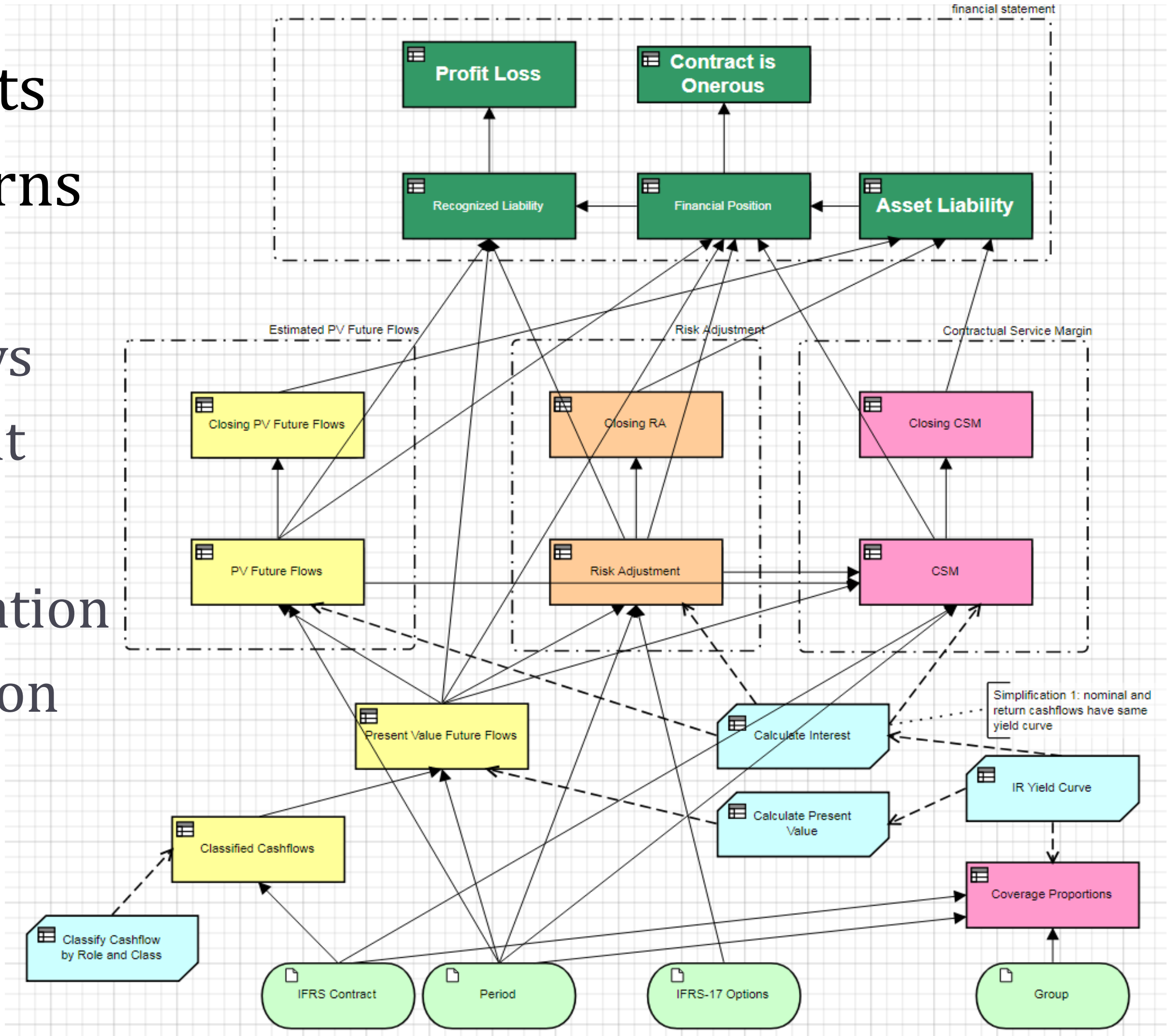
- International Financial Accounting Standard (IFRS) 17
  - Financial statement and disclosure of incremental profit
  - Determination of contractual service margins
  - Dynamic detection and handling of 'onerous' contracts
- Contract level decision, involving 100Ms contracts
- Ideal for High-Performance Decision Management
  - IFRS-17 has many options and variations
  - IFRS-17 will change regularly in the early days
  - IFRS-17 metrics can be used in many personalized decisions
    - Customer value, NBA, claim/fraud propensity, product design

Example High Performance Decision

- Decision Model Statistics
  - Modelled and tested using Trisotech DES
  - 2 Main DRDs, 6 subviews
  - 37 decisions, 29 Knowledge Sources, 6 BKMs, 4 Input Data
  - Covers about 25% of IFRS-17 including
    - Initial recognition, contract asset/liability [16(a), 32, 36, 38, 47]
    - Subsequent measurement, becoming profitable/onerous
    - Release of CSM in profitable and onerous contracts
    - Reconciliation of contract liability [44(be), B96-B97, 101]
    - Disclosure of profit and loss [48]
    - Insurance service result [84-85, 100, B123 or B124]

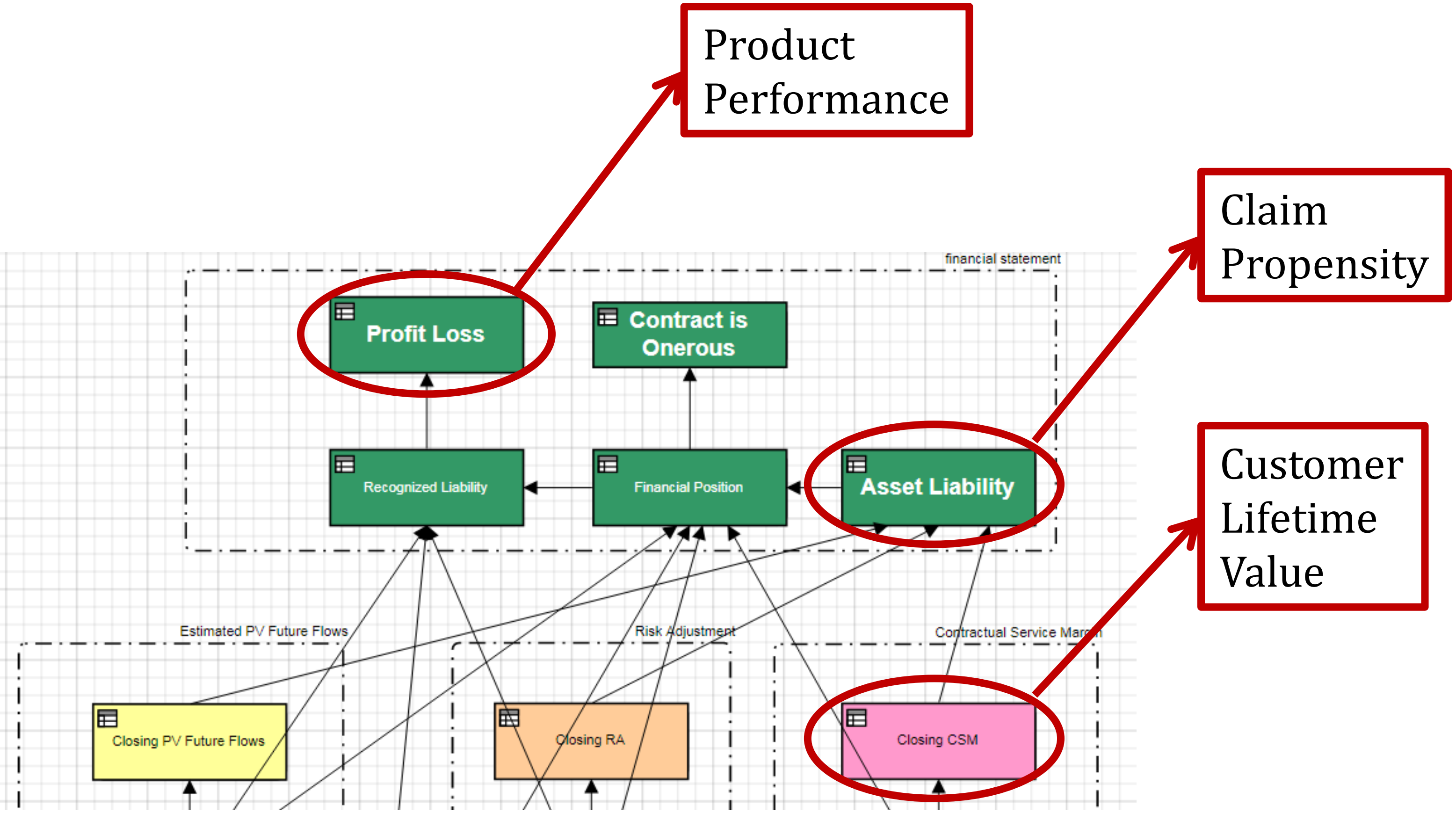
Decision Model Key Features

- Fine Grain Inputs
- Separate Concerns
  - PV Calculations
  - Fulfilment Flows
  - Risk Adjustment
  - CSM
  - Coverage Allocation
  - Financial Position
  - P&L



Decision Model Key Features

- Has Impact Outside Compliance



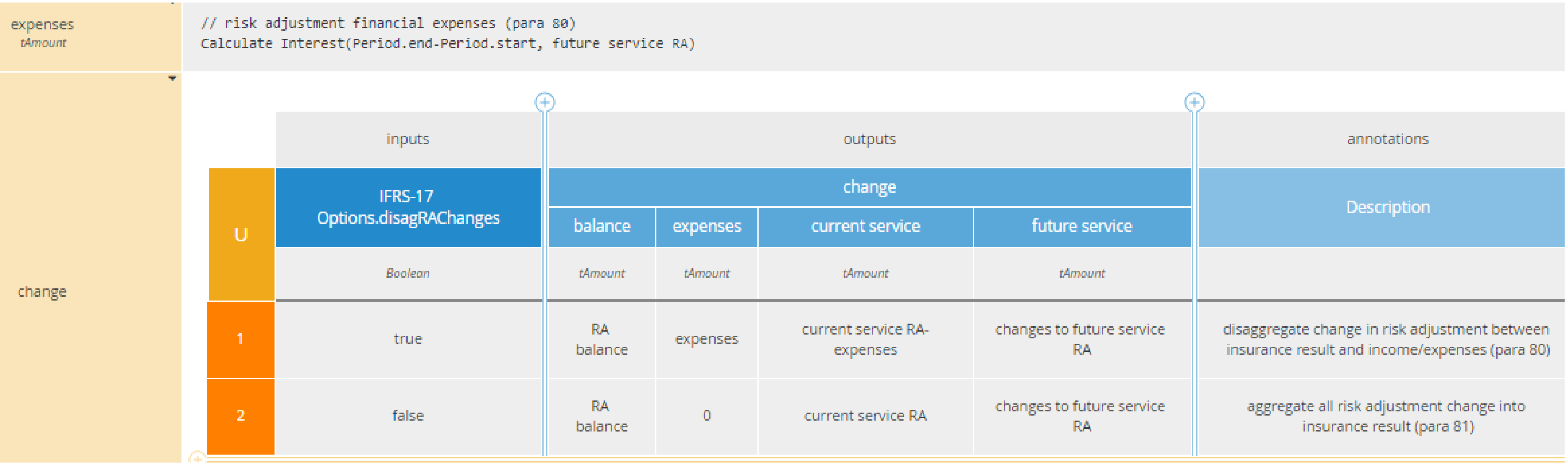
Product Performance

Claim Propensity

Customer Lifetime Value

Decision Model Key Features

- Focus on Traceability
  - For example: Risk Adjustment Disaggregation option



expenses
tAmount

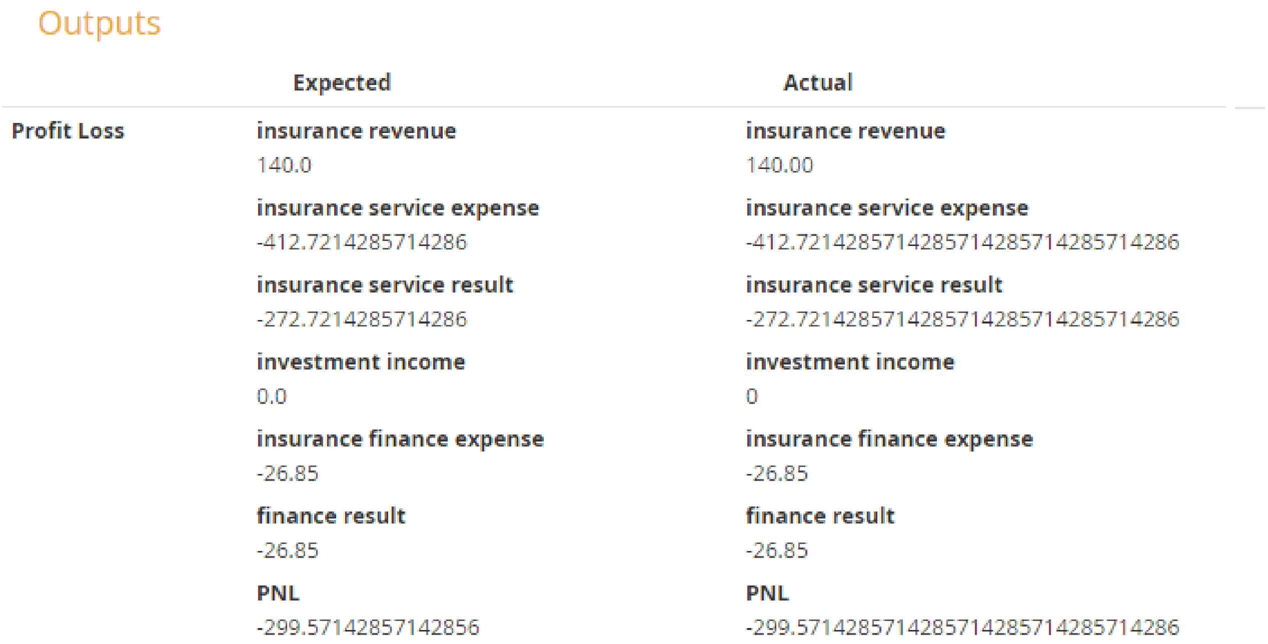
```
// risk adjustment financial expenses (para 80)
Calculate Interest(Period.end-Period.start, future service RA)
```

change

U	inputs	outputs				annotations
	IFRS-17 Options.disagRAChanges	change				Description
		balance	expenses	current service	future service	
	Boolean	tAmount	tAmount	tAmount	tAmount	
1	true	RA balance	expenses	current service RA-expenses	changes to future service RA	disaggregate change in risk adjustment between insurance result and income/expenses (para 80)
2	false	RA balance	0	current service RA	changes to future service RA	aggregate all risk adjustment change into insurance result (para 81)

Decision Model Key Features

- Focus on Test Driven Approach
  - Model supports first 50 illustrative examples with tests
  - Used in production, in training, to assess options
  - Coupled with dashboard for profit projections



Outputs

	Expected	Actual
Profit Loss	**insurance revenue** 140.0	**insurance revenue** 140.00
	insurance service expense -412.7214285714286	**insurance service expense** -412.7214285714285714285714285714286
	insurance service result -272.7214285714286	**insurance service result** -272.7214285714285714285714285714286
	investment income 0.0	**investment income** 0
	insurance finance expense -26.85	**insurance finance expense** -26.85
	finance result -26.85	**finance result** -26.85
	PNL -299.57142857142856	**PNL** -299.5714285714285714285714285714286

Coming Up…

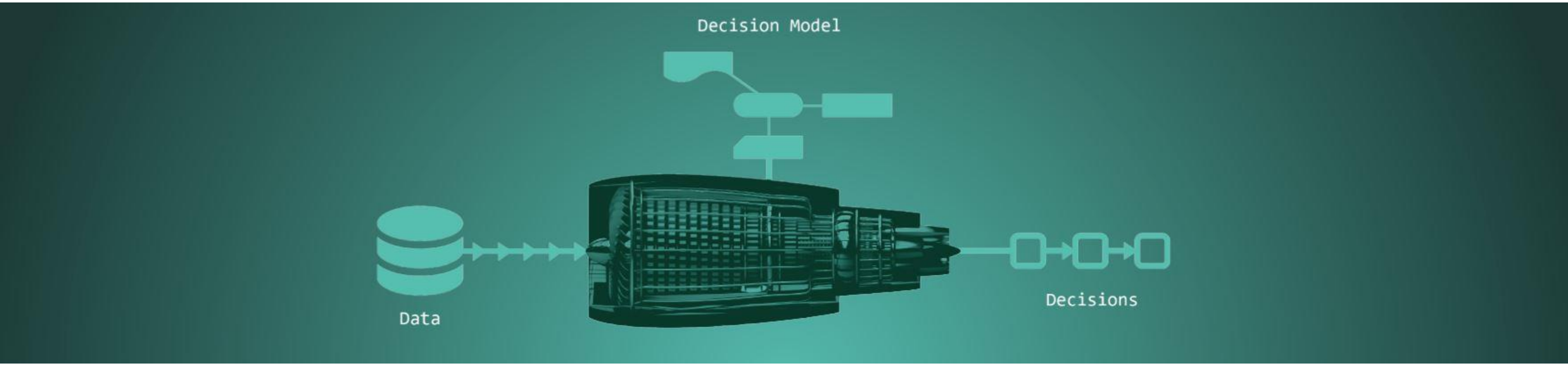
- Key Challenges in Efficient Execution of This Model
  - How to create efficient services from DMN
  - DMN's barriers to efficiency
  - Our suggestions to overcome them
- Demonstration of Model Execution Performance

Achieving High Efficiency – The RPL Language

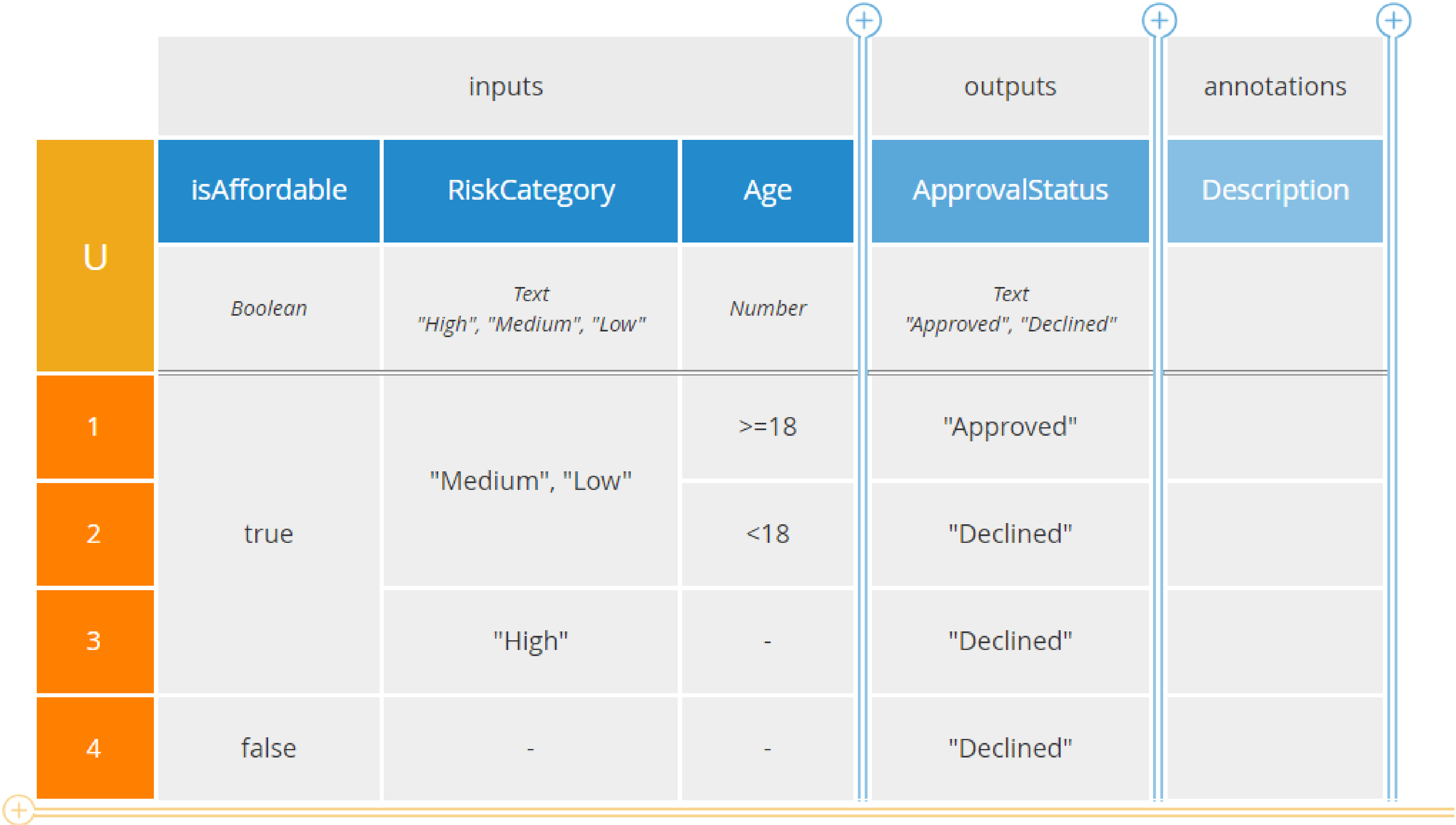
- RPL – a Simple, Low-level Language
- Compilation Directly into Machine Code
- Tiny Runtime Footprint (4Mb RTS Code)
- Logic Expressed Entirely as Decision Tables
- Multi-Ruling by Default
- Output Assignments Performed by Actions
- Built-In Iteration
- Easily Extended to Add New Features
- The RPL Compiler is Written in RPL

Achieving High Efficiency – RPL Implementation

- Condition Tests Strictly Sequential
- Rules Maintained as Bitmasks Selectively Cleared
- Some Tests Need only Two Machine Instructions
- Positive & Negative Condition Tests in Parallel
- Tests in Dead Rules can be Optimised Away
- Inline Code Wherever Possible



A Simple DMN Decision Table



	inputs			outputs	annotations
U	isAffordable	RiskCategory	Age	ApprovalStatus	Description
	Boolean	*Text "High", "Medium", "Low"*	*Number*	*Text "Approved", "Declined"*	
1	true	"Medium", "Low"	>=18	"Approved"	
2			<18	"Declined"	
3		"High"	-	"Declined"	
4	false	-	-	"Declined"	

The Same Decision Table Translated To RPL

```
*DETAB Approval_Status
I Approval_Status    I
  Approval_Status$$ I
*
* Decision table Approval_Status     | Hit Policy: UNIQUE
*
M
  isAffordable        =       $TRUE             Y Y Y N ELSE=SINGLE
  RiskCategory        =       "Medium","Low"    Y Y - - -
  RiskCategory        =       "High"            - - Y - -
  Age                 >=      18                Y N - - -
A
  $$UNIQUE_FAIL                                 . . . . X
  Approval_Status     MV      "Approved"        X . . . .
  Approval_Status     MV      "Declined"        . X X X .
  $$RETURN                    Approval_Status   X X X X .
  $$FAIL                                        . . . . X
```

Converting DMN to RPL

- 2-Stage XSL Transform Using XSLT 3.0
- FEEL Expressions Parsed Into RPL Code

  (Writing parsers in XSLT is hard!)
- RPL Language Enhanced to Support
  - FEEL date and duration types
  - DMN hit policies
  - Decimal-128 numbers

Elegance Versus Efficiency

- Single Number Type: Decimal-128
- Dynamic Strings
- Immutable Lists
- Dynamic Interim Context and Return Types
- Arcane Hit Policies (P, O & Multiple Outputs)
- Tri-State Logic (NULLs)
- Complex Conditional Expressions
- Enumerations Expressed as Strings

Single Number Type: Decimal-128

- ±0.000000000000000000000000000000000×10^{-6143} to ±9.999999999999999999999999999999999×10^{6144}
  - Universe contains about 10^{86} particles
  - World money supply is about 10^{14} US dollars
- Disadvantages
  - Very few hardware platforms implement decimal-128
  - Emulation cost on Intel is a factor of about 100
  - Integers much faster for age, count of orders etc.
  - Hard to deduce when integers can be used instead
  - Constraints? Modelling tool analysis?

Demonstration in RapidGen Genius

- Running on Mobile Core i7-4710HQ @ 2.5GHz
- No Network Connection
- Translate IFRS-17 DMN XML to RPL (Two Steps)
- Compile RPL Model Code with Test Program
- Run Test Program to Check Correct Outputs
- Run Many Silent Passes to Estimate Timing

Conclusions

- Efficiency is Still Vital
- DMN Poses Challenges to High Performance Execution
  - Solutions involve
    - Hardware aligned implementation (e.g., ‘rule masks’)
    - Performance best practices or tooling (e.g., explicit sub-typing)
    - Minimizing instruction count and re-evaluation
    - Using all available parallelism (e.g., condition tests)
    - Additions to standard
      - □ explicit integer type constraint
      - □ optional treatment of NULL
- Follow Up
  - jolyon@rapidgen.co.uk, purchase@luxmagi.com

Any Questions?