What Does a Toolchain for Automating Legislation Eventually Become?

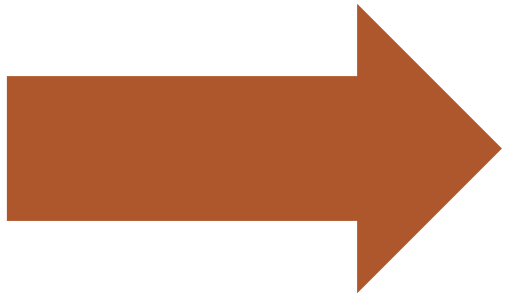
Davin Fifield – VP Product Development, Oracle

Surend Dayal – Partner, Deloitte; Senior Lecturer (Hon.), Australian National University

Don Syme – Principal Researcher, Microsoft (1998-); SoftLaw (1990-91)

Hey, let's do a programming language for the law!

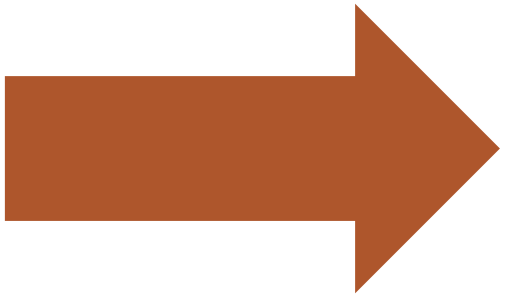
The forces of Time, Destiny, Reality, Practicality, Market, Longevity, Opportunity



???

Hey, let's do a toolchain for automating administrative law!

The forces of Time, Destiny, Reality, Practicality, Market, Longevity, Opportunity



???

1989

2022

Evolution

1989 – 1993
STATUTE Inference Machine; Rulebase Workshop

1994 – 2002
STATUTE Expert

2003 – 2008
RuleBurst; New non-RETE algorithm

2009 – 2019
Oracle Policy Automation

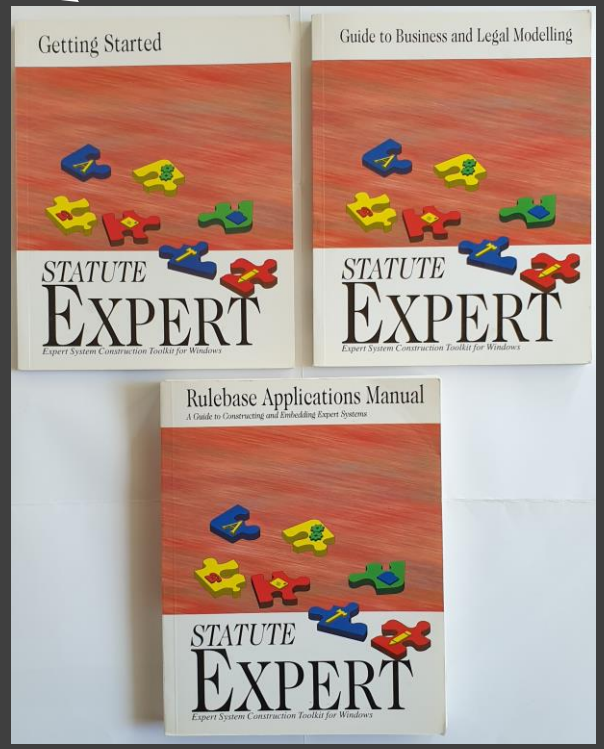
2020 – present
Oracle Intelligent Advisor

The Origins - STATUTE

1990s - STATUTE

Peter Johnson
Administrative Lawyer
"The Annotated Social Security Act 1989"

David Mead
Computer Programmer



- For **direct use** by **administrative lawyers**
- Cut the **programmers out of the loop**
- Targeted to **maintaining legislation under change**
- Knowledge represented using **limited natural language**

1990s – STATUTE Programming Model

- You're authoring a "Rulebase Application"
- The "Program" is logical **rules** in **limited natural language**
- The "IDE" helps you **build** and **maintain** this program
- The "Runtime Engine" knows how to
  - collect facts
  - manipulate limited natural language, e.g. ask questions
  - generate practical interviews , decisions, reports
- "Deployment" into Windows, later Web

Practical isomorphism – source legislation

Division 1—Eligibility for family tax benefit

Subdivision A—Eligibility of individuals for family tax benefit in normal circumstances

21 When an individual is eligible for family tax benefit in normal circumstances

(1) An individual is eligible for family tax benefit if:

(a) the individual:

(i) has at least one FTB child; or

(ii) has at least one regular care child who is also a rent assistance child; and

(b) the individual:

(i) is an Australian resident; or

(ia) is a special category visa holder residing in Australia; or

(ii) satisfies subsection (1A); and

(c) the individual's rate of family tax benefit is greater than nil.

Practical isomorphism – modelled rules

The individual is eligible for family tax benefit if

the individual satisfies paragraph 21(1)(a)

the individual has at least one FTB child

or

the individual has at least one regular care child who is also a rent assistance child

and

the individual satisfies paragraph 21(1)(b)

the individual is an Australian resident; or

the individual is a special category visa holder residing in Australia; or

the individual satisfies subsection (1A)

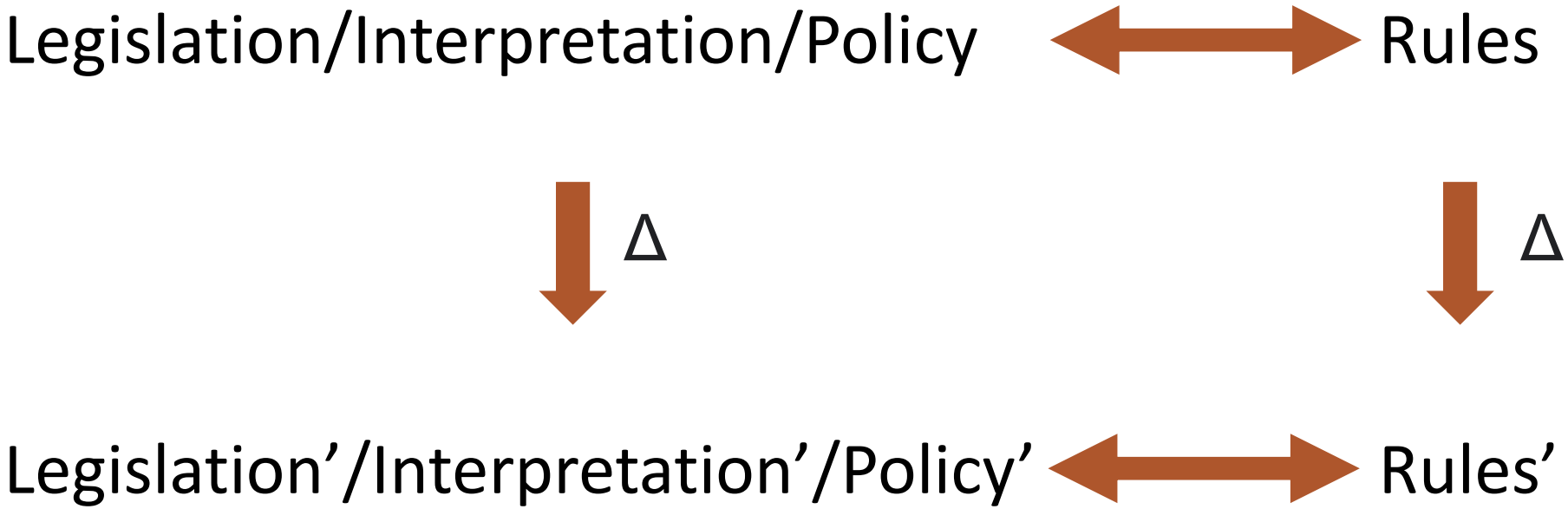
and

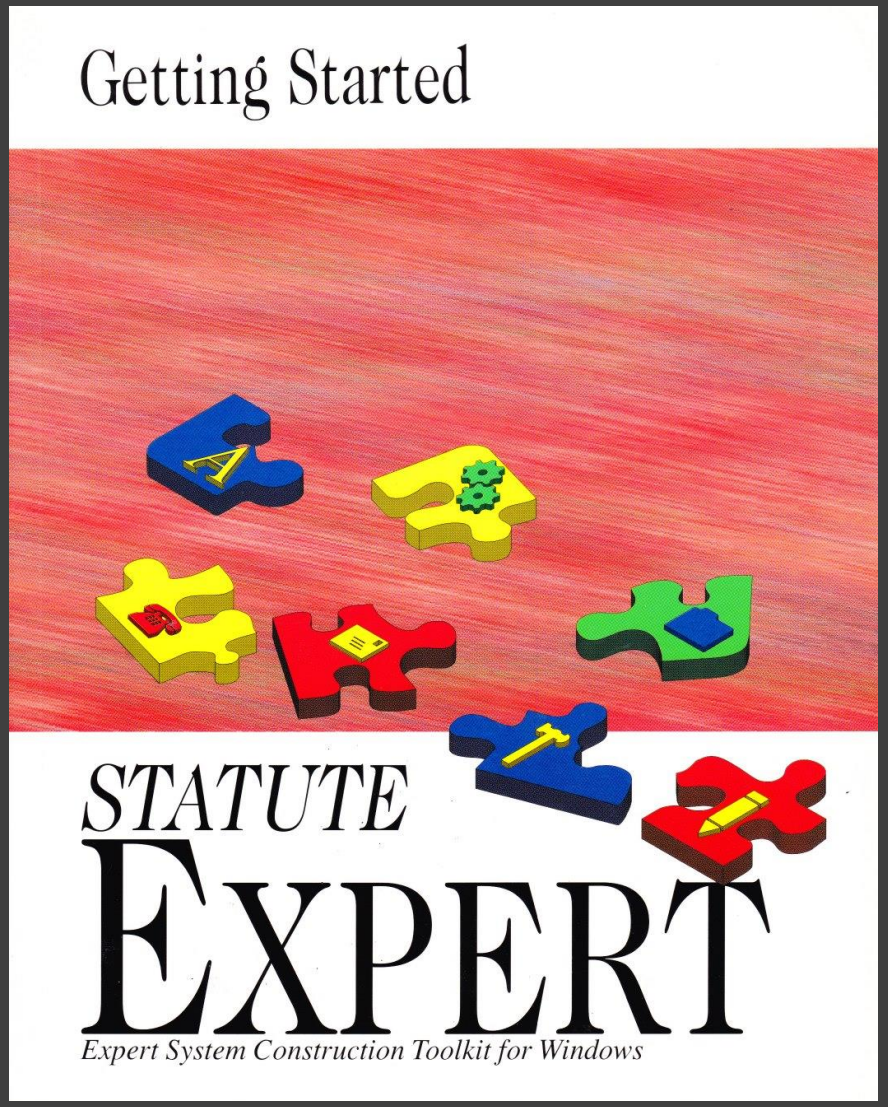
the individual satisfies paragraph 21(1)(c)

the individual's rate of family tax benefit is greater than nil

the individual's fortnightly rate of family tax benefit > 0

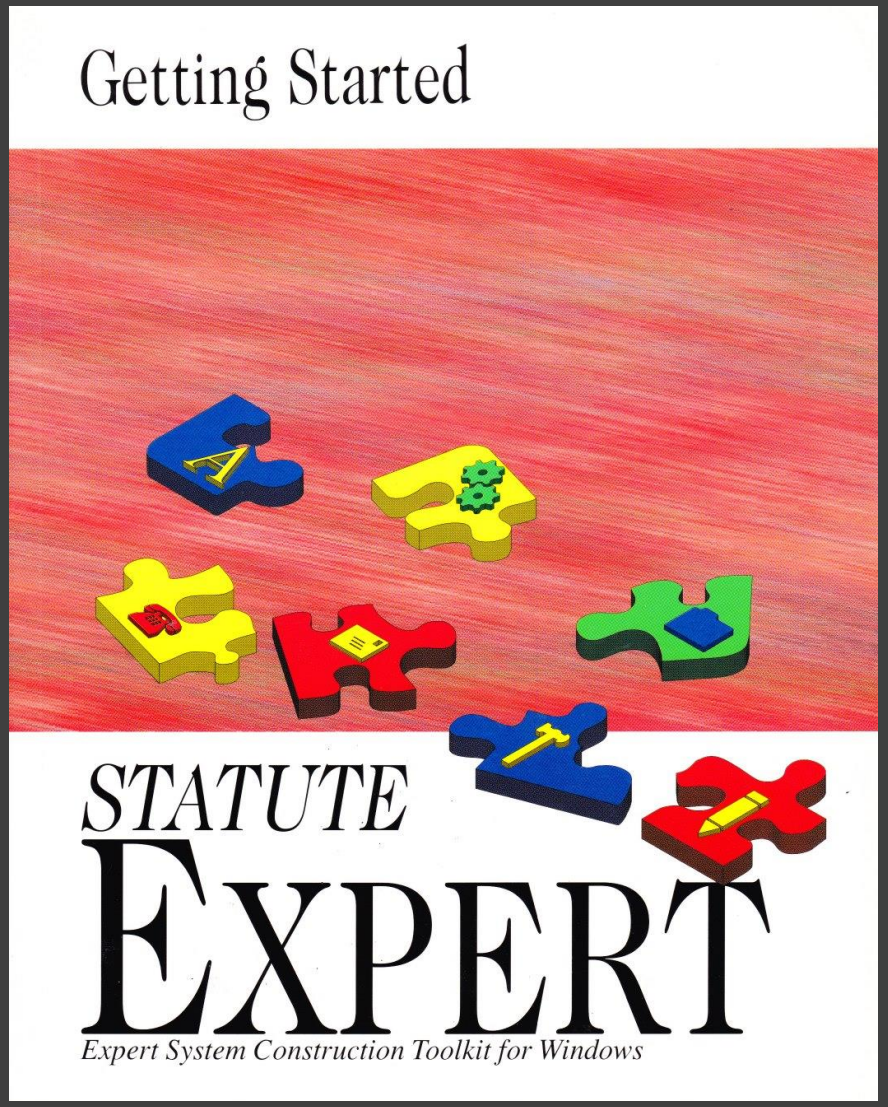
“Practical isomorphism” = maintenance under change





Contents

https://github.com/dsyme/ProLaLa2022/tree/main/STATUTE%20Expert



https://github.com/dsyme/ProLaLa2022/tree/main/STATUTE%20Expert

STATUTE Expert Methodology

- Addressed deeply practical considerations
  - Methodology for Modelling
  - Interpretation v. Legislation
  - Business Process Automation
  - Auditing Systems
- Note these topics are fundamental to any practical toolchain for automating legislation
- Rooted in deep experience with implementing administrative law

1989-1996 – STATUTE First use cases

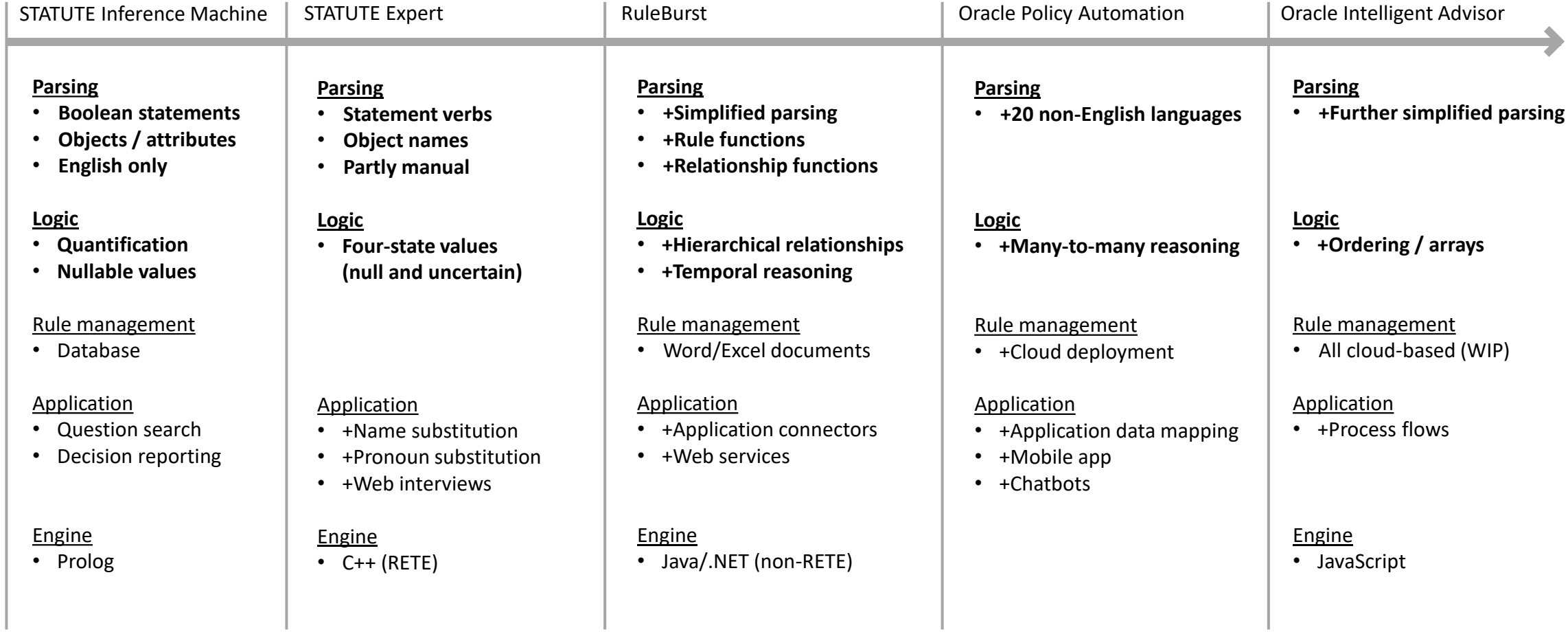
- Claims for non fault accident compensation (New Zealand ACC)
- Compensation Claims (Australian Department of Veterans Affairs)
- Social Security entitlement and calculation (Australia)
- Tax Law Guidance (Her Majesty’s Revenue and Customs, UK)

The Present – Oracle Intelligent Advisor

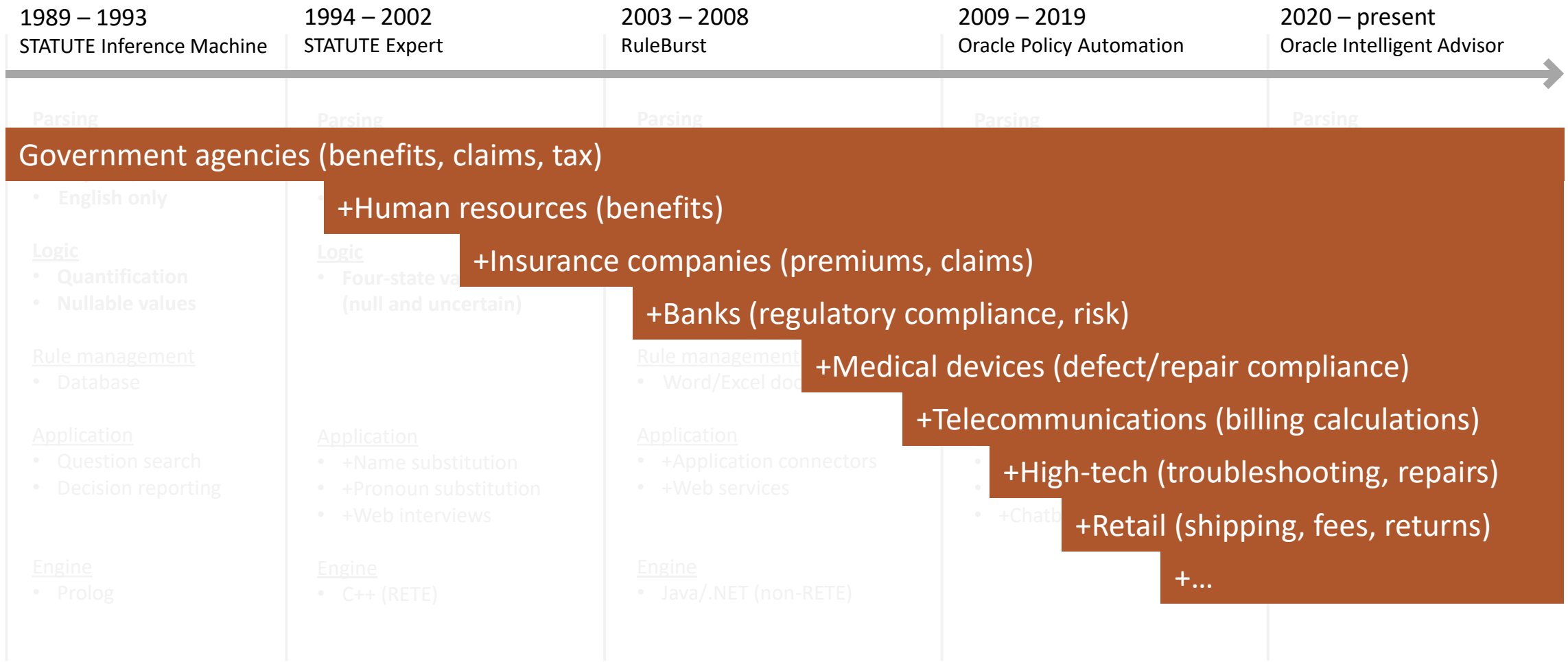
Demonstration

Oracle Policy Modeling

From a legislative toolchain to Oracle Intelligent Advisor - Technical



Use case applicability over time



Discussion

Summary : Discuss!

*Toolchains for **legislation** that find widespread adoption eventually become toolchains for **automating business policy***

Why? Because legislation-in-practice has strong affinity with business-policy-in-practice, and all the forces of time, cost and practicality lead any successful toolchain towards the latter

Toolchains for legislation must be grounded in limited natural language to be long-term successful

Why? Because legislation's primary knowledge representation is natural language, its interpretation is business policy and practical isomorphism is key to maintenance

Papers

Primary references for Oracle Intelligent Advisor

- https://bit.ly/OracleCXIA
- https://bit.ly/OracleIADoc

Primary reference for STATUTE/OIA's use of limited natural language representation

- ICAIL 1991

Primary reference for STATUTE/OIA's methodology insights

- ICAIL 1993

Links

- Oracle Intelligent Advisor – use it in the cloud!
- Archival material and paper citations: https://github.com/dsyme/ProLaLa2022
- Contact details:
  - Davin Fifield | LinkedIn; davin.fifield@oracle.com
  - Surend Dayal | LinkedIn; surend.dayal@anu.edu.au
  - Don Syme | LinkedIn, dsyme@microsoft.com

Questions - Audience

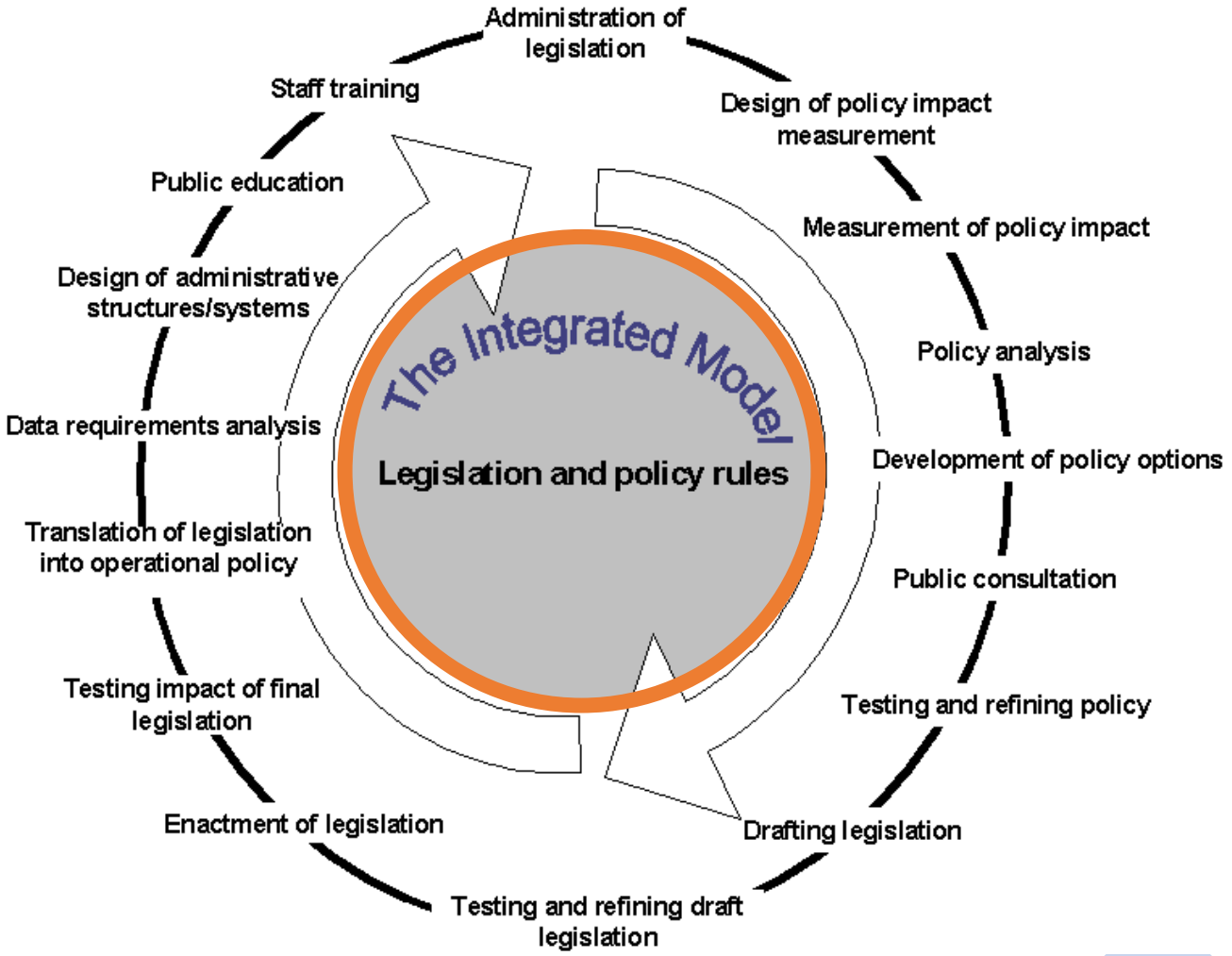
Thank you!

Davin Fifield – VP Product Development, Oracle

Surend Dayal – Partner, Deloitte; Senior Lecturer (Hon.), Australian National University

Don Syme – Principal Researcher, Microsoft (1998-); SoftLaw (1990-91)

A vision for models at the heart of the legislative lifecycle



Historical challenges to widespread adoption

- Separation between policy and administrative arms of government
- Momentum for existing legislative/policy drafting processes
- Changing the way complex rules are managed means a massive change to the way the organisation is run
- Implementation Partners prefer to code in tools they already know
- “Business technologists” were not yet a thing

Commercial realities drove broadening to any “business policy” modelling use cases

Summary #2

The history of STATUTE/OIA gives a practical guide to the commercial domains that have synergy with automating legislation

Why? STATUTE evolved into OIA under commercial pressures. This evolution reveals the areas that benefit from techniques that originated with automating legislation

Group Questions : Discuss!

What advice for a young researcher in programming languages and the law?

Is there a role for AI in the technical areas? (Parsing, Logic, Rule management, Application, Engine) How about advanced logic techniques?

How can researchers get involved in OIA today?