Deloitte.
Digital

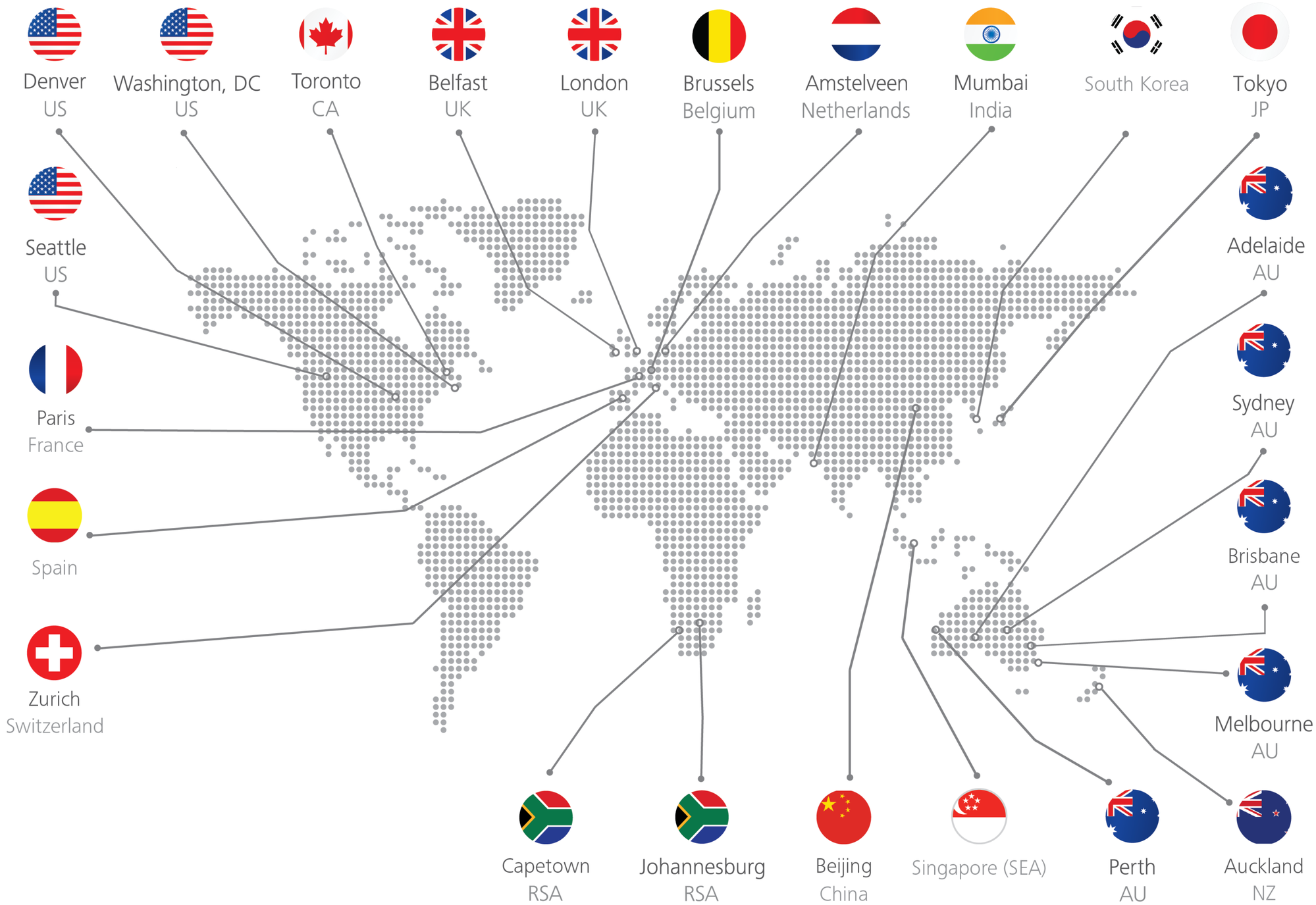
Building a high-performing decision engine

When Risk meets RETE

Andrew McKee
Deloitte Digital
2015

Who are we?

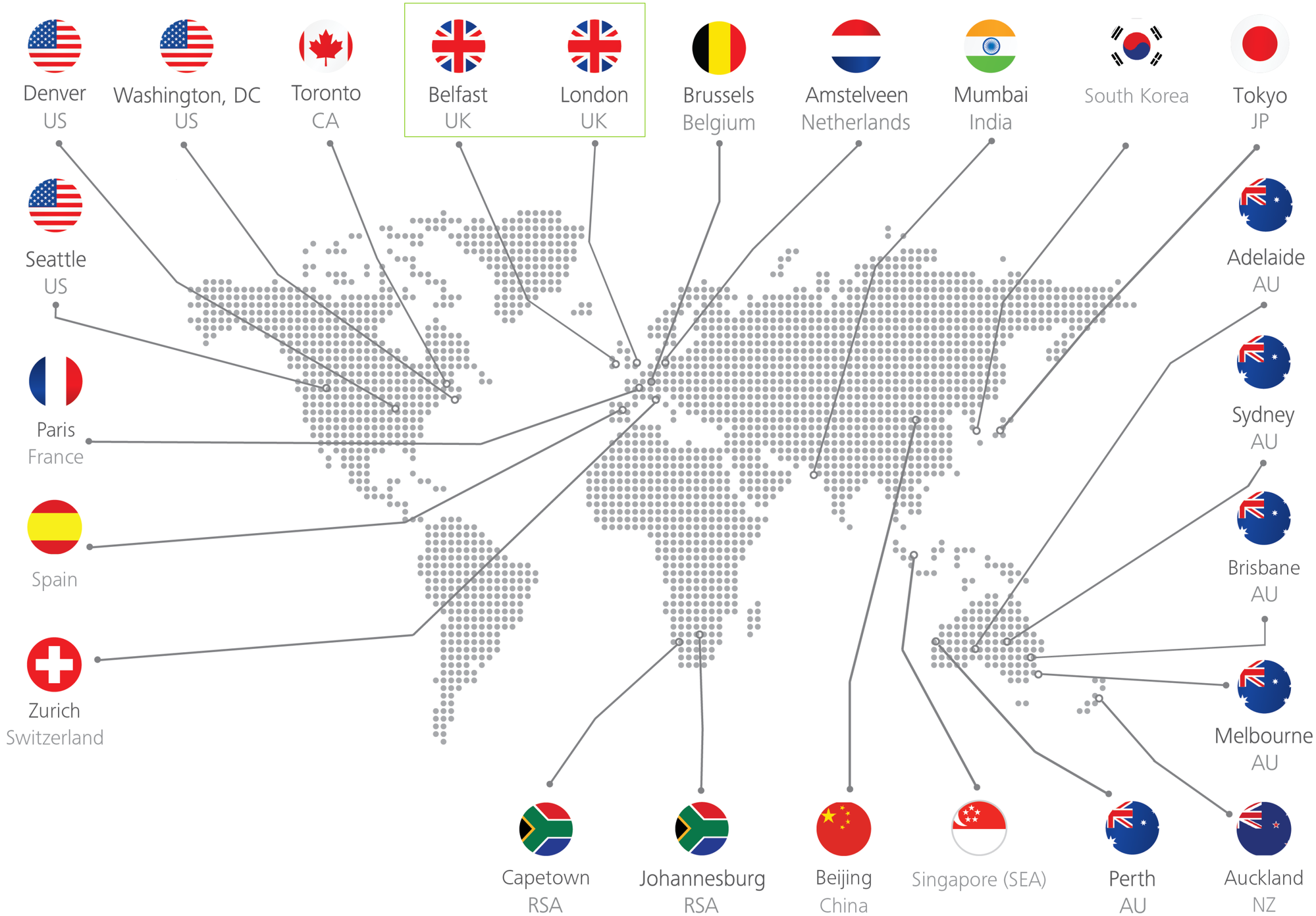

Strategy
Digital ERP
Mobile
Deloitte.
Digital
Digital content
Social
Web



23
STUDIOS

10
TIME ZONES

05
CONTINENTS



23
STUDIOS

10
TIME ZONES

05
CONTINENTS

Who am I?

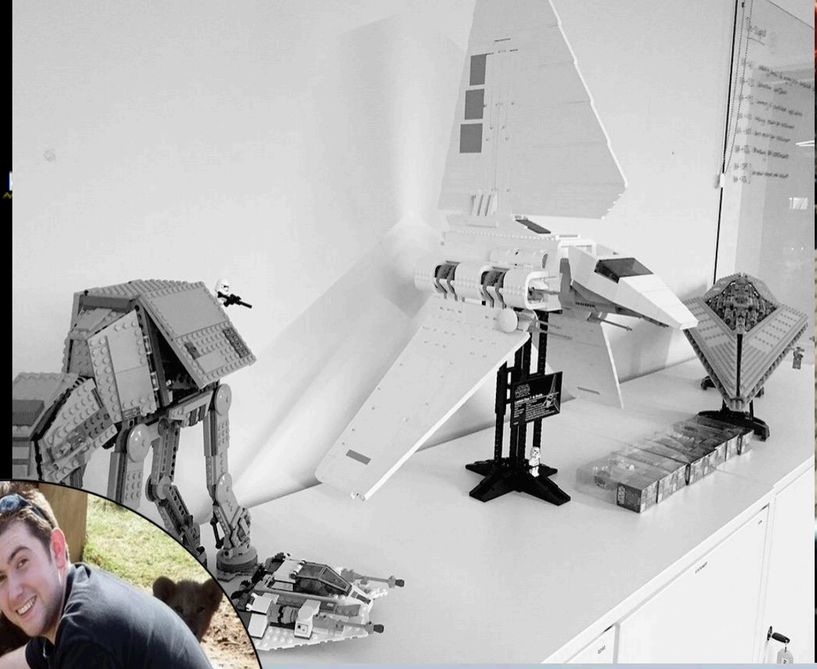
```
 * @param facts           the fact base to assert on
 * @param eventListeners Required event listeners
 * @param ksession       Session to act upon
 * @throws RuleServiceException
 */
protected void fireAllRulesInKnowledgeBase(Map<String, Object> globals, Map<String, Object> facts,
                                           List<WorkingMemoryEventListener> eventListeners, StatefulKnowledgeSession

    try {
        // fire rules, and obtain a handle to the inserted facts
        final Map<String, FactHandle> factHandles = fireRulesWithSession(globals, facts, eventListeners, ksession);

        // retrieve facts and return
        for (String factName : factHandles.keySet()) {
            final FactHandle fh = factHandles.get(factName);

            // retrieve fact if we have a handle for it (will be null if no handle)
            Object retrievedFact = null;
            if (null != fh) {
                retrievedFact = ksession.getObject(fh);
            }

            // update the object in the *ORIGINAL* map instance
```



Andrew McKee
Consultant
@amckee23

Our Challenge.

What was the big deal?

- In depth risk assessment was critical for the client

- Complex workflows are causing headaches for staff

- No ability to modify existing workflows

- One assessment alone had **523** individual decisions

What did we look at?

- Bespoke build
- Survey tools
- Rule Engines

What did we look at?

- Bespoke build
- Survey tools
- Rule Engines

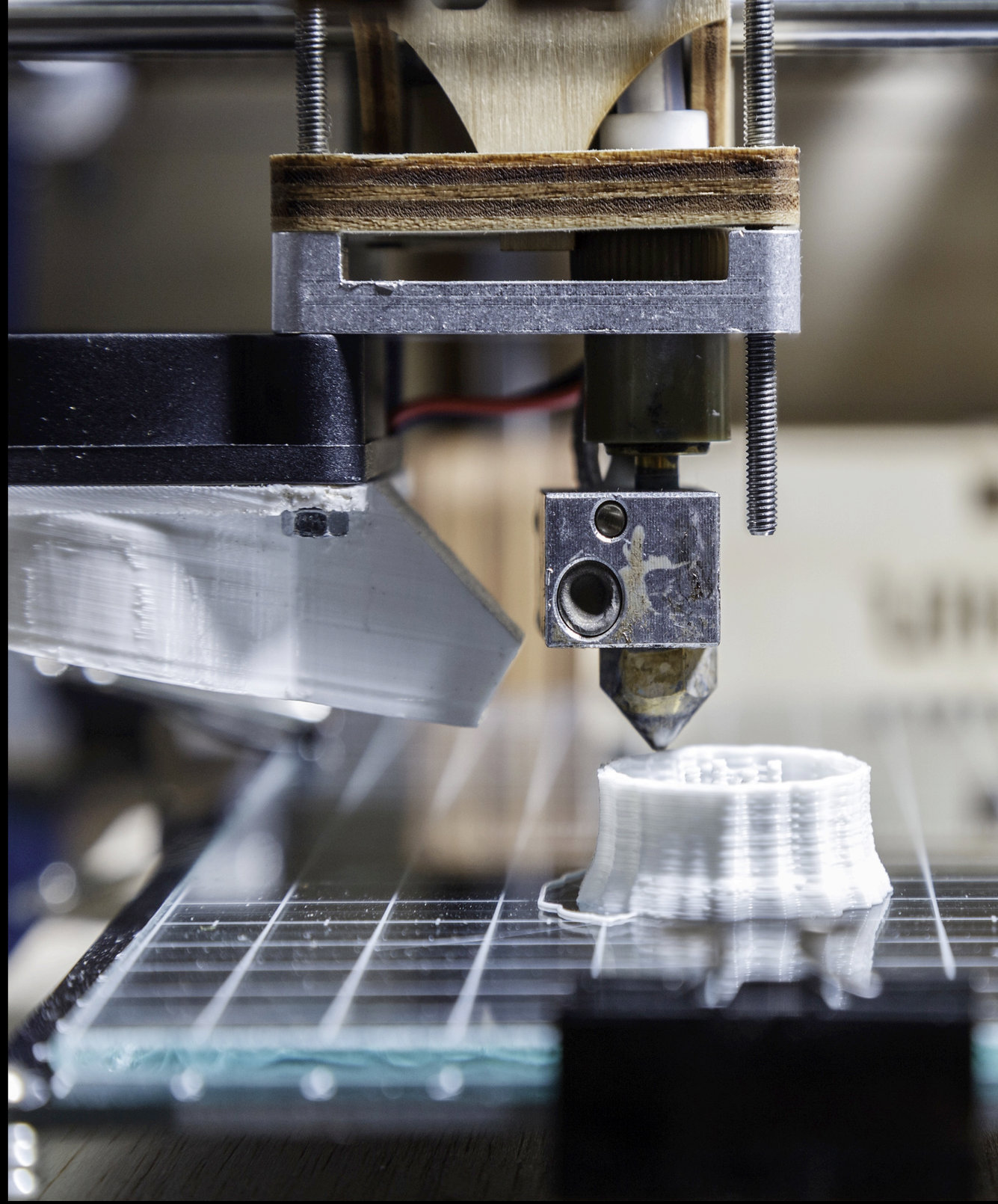
Answer: Rule Engines

What did we look at?

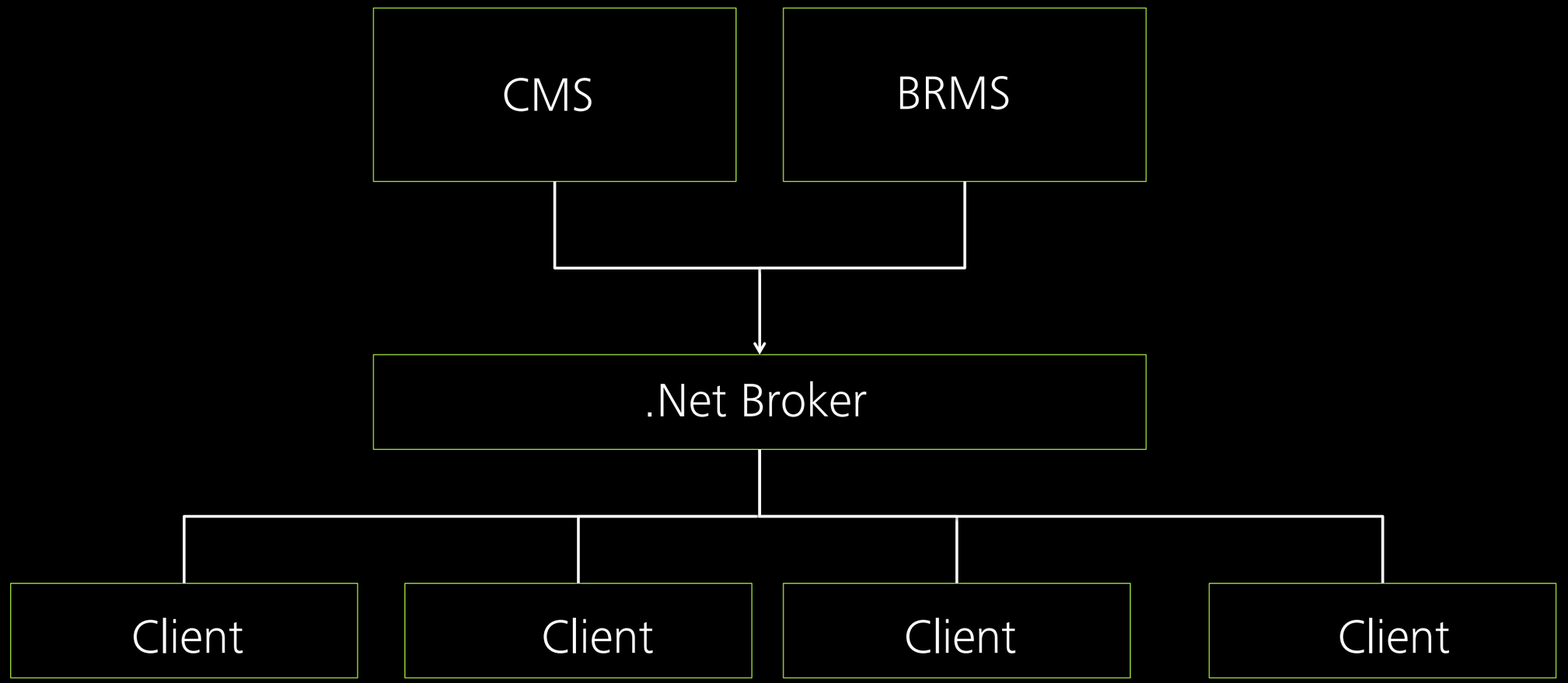
- IBM Ilog (JRules)
- Drools .Net
- BRMS

Why BRMS?

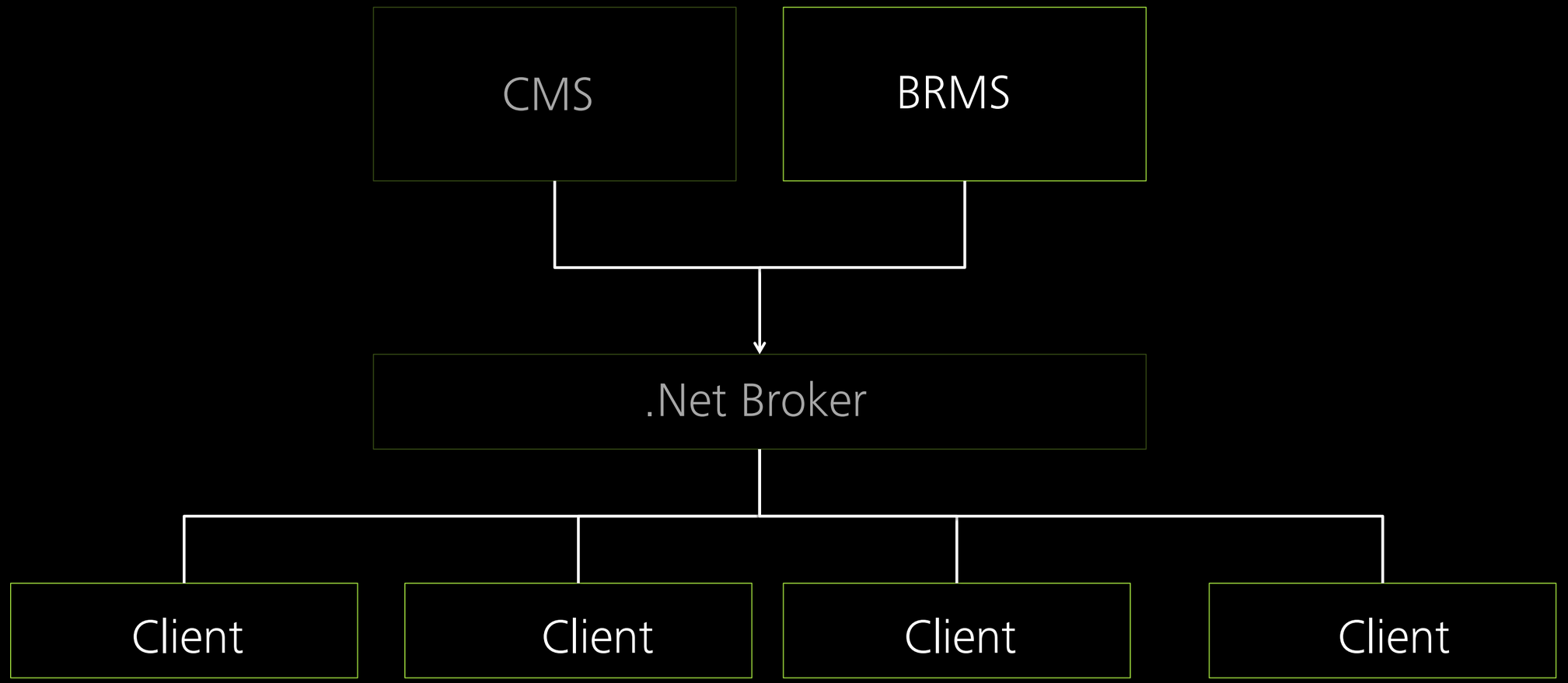
- Tohu for inspiration
- Performance
- Customization
- Domain specific language



What did we build?



What am I going to talk about?



The language

```
rule "add broken build over weekend risk if the build is broken on a Friday"
    when
      Question “Is-The-Build-Broken" has an answer of True
      Question “What-Day-Of-Week-Is-It" has an answer of "Friday"
    then
        Add risk “Broken-Build-Over-Weekend"
        Show question “Will-You-Fix-Build"
end
```

The language

Question What day of the week is it has an answer of Friday

The language

[condition][]Question "{id}" has an answer of {answer}

The language

[condition][]Question "{id}" has an answer of {answer}=Question(controlId == "{id}", hasAnswer == **true**, answer == {answer})

The language

- Iteration was key
- Spend time with the end user
- Understand how the user talked about risk
- End result? Better adoption

[condition][]Question "{id}" has an answer of {answer}=Question(controlId == "{id}", hasAnswer == **true**,

[condition][]Question "{id}" answer is **not** {answer}=Question(controlId == "{id}", hasAnswer == **true**, ans

[condition][]Question "{id}" has **not** been answered or answer is **not** {answer}=**not** Question(controlId ==

[condition][]Question "{id}" has **not** been answered or has an answer of {answer}=**not** Question(controll

[condition][]Question "{id}" list of answers **contains** "{answer}"=Question(controlId == "{id}", "{answer}"

[condition][]Question "{id}" has **not** been answered or list of answers **contains** {answer}=**not** Question(co

[condition][]Question "{id}" list of answers does **not** contain {answer}=Question(controlId == "{id}", hasA

[condition][]Question "{id}" has **not** been answered or list of answers does **not** contain {answer}=**not** Qu

[condition][]Question "{id}" answer is **true**=Question(controlId == "{id}", answer == **true**)

[condition][]Question "{id}" answer is **false**=Question(controlId == "{id}", answer == **false**)
[condition][]Question "{id}" has **not** been answered or answer is **false**=**not** Question(controlId == "{id}",
[condition][]Question "{id}" has **not** been answered or answer is **true**=**not** Question(controlId == "{id}", a
[condition][]Question "{id}" is **not** shown=**not** Question(controlId == "{id}")
[condition][]Question "{id}" is shown=Question(controlId == "{id}")
[condition][]Question "{id}" does **not** have answer=Question(controlId == "{id}", hasAnswer == **false**)

Demo Time

Play as you go (don’t hate me for the WiFi)

http://app-amckee.rhcloud.com/

How does it all work?

User answers question

Load existing state from DB

Add existing state to rule session

Fire Rules and listen to events

Add events as show / hide operations

Save new assessment state

Return new events to the user

What did we learn?

- Managing large rule sets is hard
- Avoid being too generic
- Don’t use Statefull sessions, they hurt!
- Explore rule substitution rather than authoring

What did we learn?

- Test, test and test some more
- Even with custom domain languages never underestimate users ability to break your product
- Think of coupling and cohesion when writing your rules
- Use common sense

What would we do differently?

- Add a custom authoring UI
- BRMS is awesome but make sure it’s the right tool for the job
- Start with simple then make it enterprise

Questions?

Deloitte.

Digital