# IDEal: A Legal Development Environment

**Australian Submission**

2020

Mamta Thaker — Joanna Chen — Mirhady Dorodjatun — Joshua Fourie

# Contents

# Part I

# Business

# Part II

# Technical

A legal matter processed in IDEal traverses four states:

- *Generation* ($\mathcal{G}$).
- *Representation* ($\mathcal{R}$).
- *Transformation* ($\mathcal{T}$).
- *Presentation* ($\mathcal{P}$).

We can understand IDEal as a system of *plug-ins* which either generate, or commonly access and transform a unified representation of a legal matter into derivative states. Consequently, the system is a series of machines mapping $[\ \alpha_i \in \mathcal{G}\ ] \to \mathcal{R}$, or $\mathcal{R} \to [\ \alpha_i \in \mathcal{T} \cup \mathcal{P}\ ]$. We denote $\langle \alpha_i \rangle$ as a *generator* state which can generate $\mathcal{R}$, and $\alpha_i$ as a *producible* state which be produced by some action on $\mathcal{R}$.

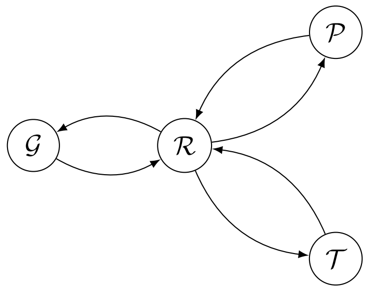



Figure 1: Visualising the interactions between the states.

In this section, we define $\mathcal{R}$, and provide an example of $(\mathcal{G} \to \mathcal{R})$ and $(\mathcal{R} \to \mathcal{P})$.

## 1 Defining the Representation

Our goal is to define a mathematical structure for $\mathcal{R}$ which encodes legal information and maximises the number of producible states. Given that the representation is driven by the encoded legal information, we begin by discussing our generalisation of information relevant to a legal case.

### 1.1 Generalising the Information in a Legal Case

Generally, a lawyer functions as a mechanism for identifying the *existence* or *non-existence* of a legal relationship between *objects*, as well as the extrapolation of any implications for a client.

**Facts, Nodes and the NodeState.** We describe a legal object as a `Node`, and define the *factual* actions, attributes or character of the `Node` as the `NodeState`. The $\texttt{NodeState}_n$ is a set of $n$ `Fact` objects, $\texttt{NodeState}_n = \{ \texttt{Fact}_0, .., \texttt{Fact}_n \}$, which are generated and edited by an oracle, $\mathcal{F}$:

$$\mathcal{F} : [\ \{ \texttt{Fact}_0, .., \texttt{Fact}_n \} \rightarrow \texttt{NodeState}_n\ ] \tag{1.1}$$

$$\mathcal{F} : [\ (\texttt{Fact},\ \texttt{NodeState}_n) \rightarrow \texttt{NodeState}_{n+1}\ ] \tag{1.2}$$

$$\mathcal{F} : [\ \texttt{NodeState}_n \rightarrow \texttt{NodeState}_{n-1}\ ] \tag{1.3}$$

**Sources of Law.** A `SourceOfLaw` defines the conditions, attributes or characteristics which are required in order to generate the `Role` (1.1.1) associated with a `NodeState`. Concretely, these are objects such as legislation or common law which have the inherent capacity to generate legal rights or obligations.

### 1.1.1 The Role Object

A `Role` defines the *legal personality* of a `Node` by capturing the attributes, actions, or characterstics attributable under law. A `Node` can be subject to multiple `Role` objects of arbitrary complexity, provided they are distinct under (1.6). The `Role` associated with a `Node` is generated by a pair `(NodeState, SourceOfLaw)` under the oracle function, $\mathcal{F}$, and is transformable under the `Consequence` of a `Link` (1.1.2):

$$\mathcal{F} : [\ (\texttt{NodeState, SourceOfLaw}) \rightarrow \texttt{Role}\ ] \tag{1.4}$$

$$\texttt{Consequence} : [\ \texttt{Role} \rightarrow \texttt{Role'}\ ] \tag{1.5}$$

**Equivalence of Roles.** We define an equivalence relation on a pair $(\texttt{Role}_i,\ \texttt{Role}_j)$ by comparing their generating states, such that they are only pairwise distinct where the generative facts or law diverge:

$$[\ \texttt{Role}_i = \texttt{Role}_j\ ] \iff [\ (\texttt{NodeState}_i \iff \texttt{NodeState}_j) \wedge (\texttt{SourceOfLaw}_i \iff \texttt{SourceOfLaw}_j)\ ] \tag{1.6}$$

**Role Composition.** The `NodeState` of a `Node` can generate multiple `Role` objects *iff* the (`NodeState, SourceOfLaw`) pair are distinct under (1.6). A `Role` is *reducible* where a subset of the generative pair (`NodeState, SourceOfLaw`) can produce another distinct `Role`:

$$N_n := \{ \texttt{Fact}_0, .., \texttt{Fact}_n \} \tag{1.7}$$

$$[\ \exists\ N' \subset N_n\ ] \implies [\ N_n \text{ is reducible }] \tag{1.8}$$

A `Role` which is reducible is an *extension* or *composition* of another `Role`, and the `Role` objects which are extended are called the *components of the extension.* The extended `Role` will automatically import any components of the extension objects into its own definition. We denote an extension using subset notation, such that the following indicates $\texttt{Role}_i$ is an extension of $\texttt{Role}_j$: $\texttt{Role}_j \subset \texttt{Role}_i$.

**Role Substitution.** Given that an extension implies any components, a `Node` with multiple `Role` objects *may* replace any `Role` with an extension:

$$(\ \texttt{Role}_j \subset \texttt{Role}_i\ ) \implies (\ \texttt{Role}_i \implies \texttt{Role}_j\ ) \tag{1.9}$$

### 1.1.2 Links and Consequences

A `Link` is a directed, pairwise relationship between a source, $\texttt{Role}_i$, and a destination, $\texttt{Role}_j$, which has been generated by a `SourceOfLaw`. Given a pair $(\texttt{Role}_i, \texttt{Role}_j)$ and an associated $\texttt{Link}_{i\to j}$ drawn by the oracle, $\texttt{Role}_j$ is mapped into $\texttt{Role}_{j'}$ under some `Consequence`:

$$\mathcal{F} : (\texttt{SourceOfLaw},\ \texttt{Role}_i,\ \texttt{Role}_j) \to [\ \texttt{Link}_{i\to j}\ ] \tag{1.10}$$

$$\texttt{Link} \implies [\ \texttt{Consequence} : \texttt{Role}_j \to \texttt{Role}_{j'}\ ] \tag{1.11}$$

**Example.** Consider a contract, $\mathcal{C}$, between Alice, $\mathcal{A}$, and Bob, $\mathcal{B}$. We define $\mathcal{C}$ as a `SourceOfLaw`, and both $\mathcal{A}$ and $\mathcal{B}$ as parties to the contract. Consequently, there are two links: $\texttt{Link}_{\mathcal{A}\to\mathcal{C}}$ and $\texttt{Link}_{\mathcal{B}\to\mathcal{C}}$. The `Consequence` of any `Link` with $\mathcal{C}$ is that both $\mathcal{A}$ and $\mathcal{B}$ are bound by the contract, and whatever `Role` objects they possess are modified accordingly. Furthermore, the `Role` assigned to $\mathcal{A}$ and $\mathcal{B}$ as 'parties to the contract' is an extension of the 'legal person' role, and it is therefore implied under (1.10).

**Compliance and Deviation.** A lawyer constructs a `Link(Consequence)` by analysing the `Role` of an object relative to a set of legal conditions. We define both compliance and deviation as:

$$[\ \exists\ \texttt{Link(Existence, Consequence)}\ ] \wedge \begin{cases} [\ \texttt{Role} \implies \texttt{Link(..)}\ ], & \text{Compliance} \\ [\ \texttt{Role} \not\Longrightarrow \texttt{Link(..)}\ ], & \text{Deviation} \end{cases} \tag{1.12}$$

**Example.** Continuing the previous example, we define compliance as Bob enforcing the terms of C against Alice, because Alice fulfils the relevant `Role`. Bob could not, however, enforce C against a third party, unless they also fulfilled a role which generated a `Link` to C.

# 2 Appendix A

We discuss applications of the selected representation to cases.

## 2.1 *Belgrave Nominees v Barlin-Scott Airconditioning (Aust.)*

These are the objects involved in the *Belgrave Nominees v Barlin-Scott Airconditioning (Aust.)* case:

**Example 2.1** *A Simple Relationship.*

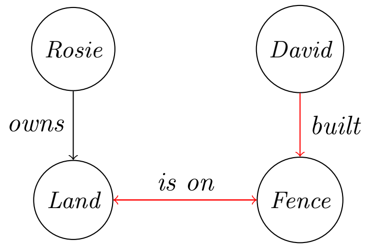
Consider the following: *David builds a fence on Rosie's land without consent.*

*From:* `Rosie`

*To:* `Land`

*Role:* `SuperiorPossessor`

*Link:* `(Yes, Modifier)`




We have implicitly assigned *Rosie* the `Role` of superior possessor of the parcel of land. Consequently, we set `Existence` := `Yes`, and `Consequence` := `Modifer`.

## 2.2 Mathematical Abstraction of a Legal Case

We assume that a lawyer is responsible for (1) identifying the existence or non-existence of *legal relationships*, and then (2) drawing conclusions about the implications of those relationships for a client. Their expertise is in determining the relevance of a particular relationship to a given case, where the relevance is guided by the severity or possibility of any consequence to a client.

## 2.3 A Starting Point

Suppose that there exists a universe of objects:

$$\mathcal{U} := \{ \ \omega_i : \omega_i \text{ is an object in the universe } \} \tag{2.1}$$

Within the universe, an object may be in a directed, pairwise relationship with another object:

$$\exists \ \omega_i : (\omega_i \ \sigma \ \omega_j) \tag{2.2}$$

We define any relationship as transitive:

$$(\omega_i \to \omega_j \ \sigma \ \omega_k) \implies (\omega_i \ \sigma \ \omega_k) \tag{2.3}$$

We also define each object discretely and uniquely:

$$[ \ (\omega_i \ \sigma \ \omega_j) = (\omega_k \ \sigma \ \omega_\ell) \ ] \implies [ \ (\omega_i \iff \omega_k) \wedge (\omega_j \iff \omega_\ell) \ ] \tag{2.4}$$

**Equivalence.** We define an equivalence relation on any machine acting on a given representation:

$$[ \ M_i(\mathcal{R}) = \alpha_i \ ] \wedge [ \ \alpha_i = M_j(\mathcal{R}) \ ] \implies [ \ M_i \Leftrightarrow M_j \ ] \tag{2.5}$$

However, for the pair $(\mathcal{R}_i, \mathcal{R}_j)$:

$$[\ M_i(\mathcal{R}_i) = \alpha_i\ ] \wedge [\ \alpha_i = M_j(\mathcal{R}_j)\ ] \ \not\Rightarrow\ [\ M_i \Leftrightarrow M_j\ ] \vee [\mathcal{R}_i \Leftrightarrow \mathcal{R}_j. \tag{2.6}$$