From Contracts to Runtime Verification

Gordon J. Pace

Department of Computer Science
University of Malta

March 2017

With a little help from my friends – Shaun Azzopardi, María-Emilia Cambronero, John Camilleri, Christian Colombo, Albert Gatt, Stephen Fenech, Luis Llana, Cristian Prisacariu, Anders Ravn, Mike Rosner, Fernando Schapachnik, Gerardo Schneider.

But what is a contract?

- Contracts as agreements between parties that accept to behave in a particular manner, with repercussions if they do not.
- Contracts vs. Properties:
  - Contracts are multi-party agreements[1].
  - Contracts can be broken.
  - Consequences of breaking a contract is itself regulated.

[1] "*Nudum pactum* — an agreement to do or to pay any thing on one side, without any compensation to the other, is totally void in law"

But what is a contract?

- Contracts as agreements between parties that accept to behave in a particular manner, with repercussions if they do not.
- Contracts vs. Properties:
  - Contracts are multi-party agreements[1].
  - Contracts can be broken.
  - Consequences of breaking a contract is itself regulated.
- For example:
  - A naked property is not a contract...
  - But a runtime monitoring script which identifies properties for different agents in a system and code to execute (reparations, compensations or enforcement) when the property is violated can be seen as one (in a certain sense).

[1] "*Nudum pactum* — an agreement to do or to pay any thing on one side, without any compensation to the other, is totally void in law"

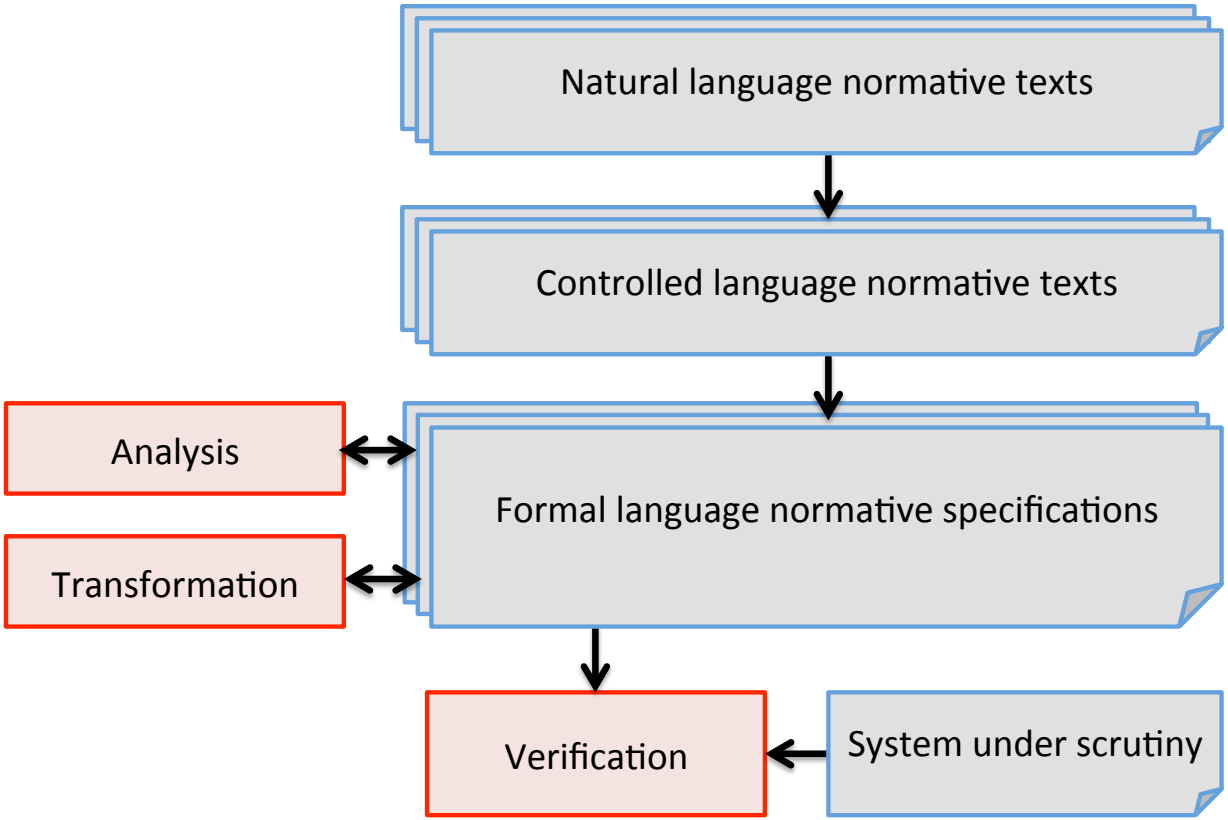
Let's step back

- Contracts are instances of *normative texts* — text identifying *ideal* as opposed to *actual* behaviour.
- In the rest of the talk I will be taking this broader view to include both contracts and legislation.

What I will be talking about

- Different approaches to formalising semantics of normative texts.
- Formal analysis of contracts.
- Natural language techniques for normative texts.
- Some real life use cases of normative text analysis.
- Where does runtime monitoring/verification/enforcement fit into this?

The obligatory diagram

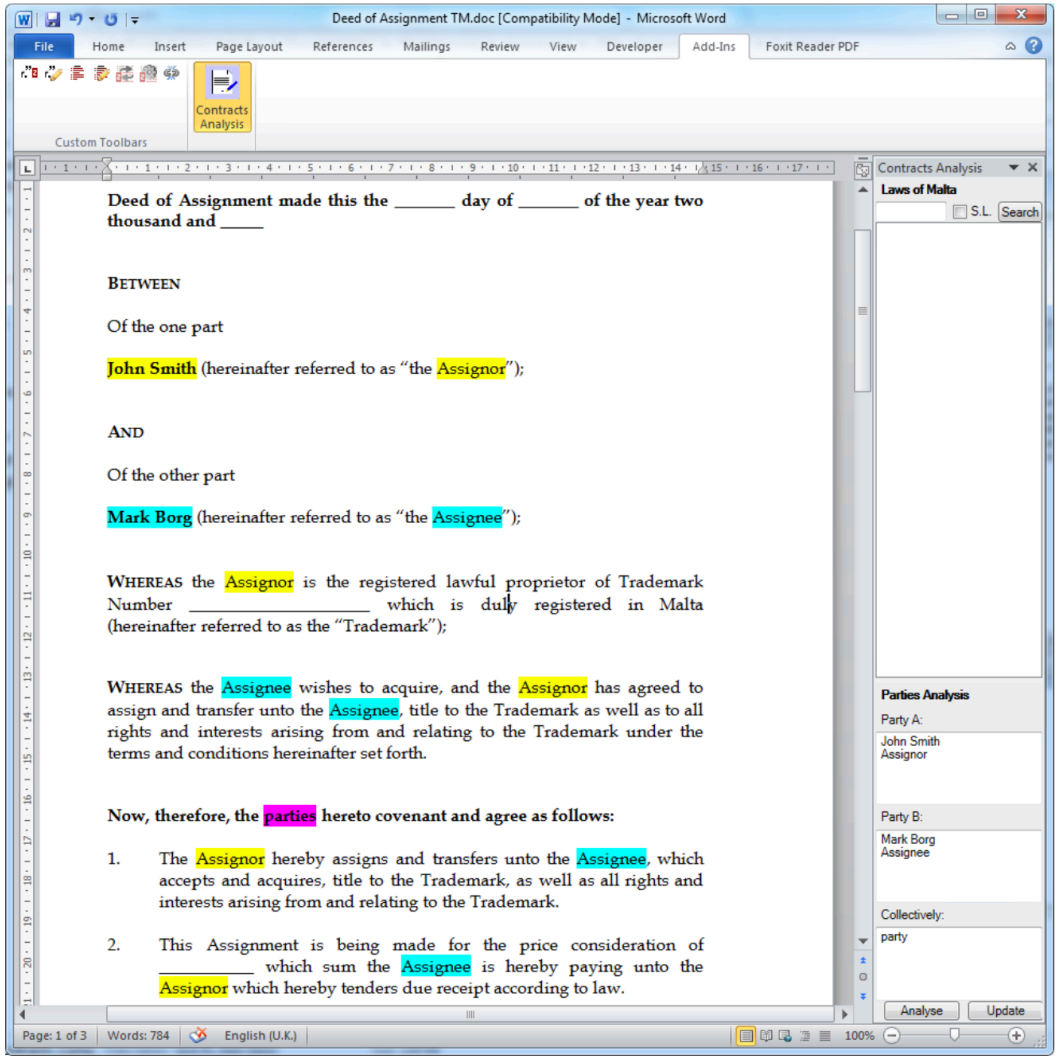


Natural language normative texts

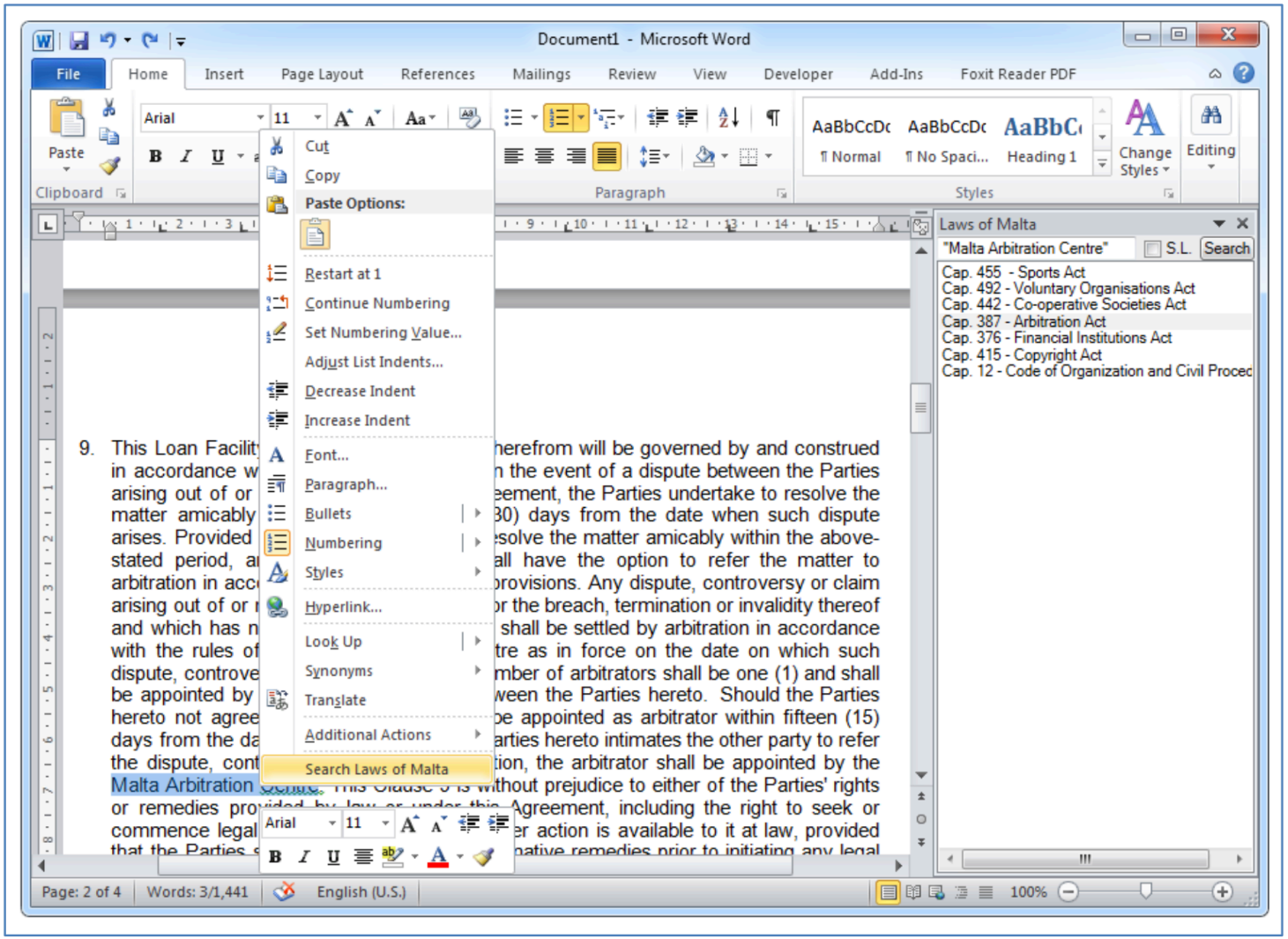
Natural Language Normative Texts

- Legalese typically uses a rather stylistically technical, yet not, prescribed language.
- Various classes of texts
  - Contracts
  - Laws/Bills/Legal notices
  - Directives
  - Court decisions
- Structurally amenable to statistical NL techniques e.g. for MT, but semantically challenging.
- Information extraction techniques work reasonably well on such texts.
- Most work limited to support intelligent browsing of contract clauses and identify links between legal texts.
- Constructing a formal model is still too unreliable.

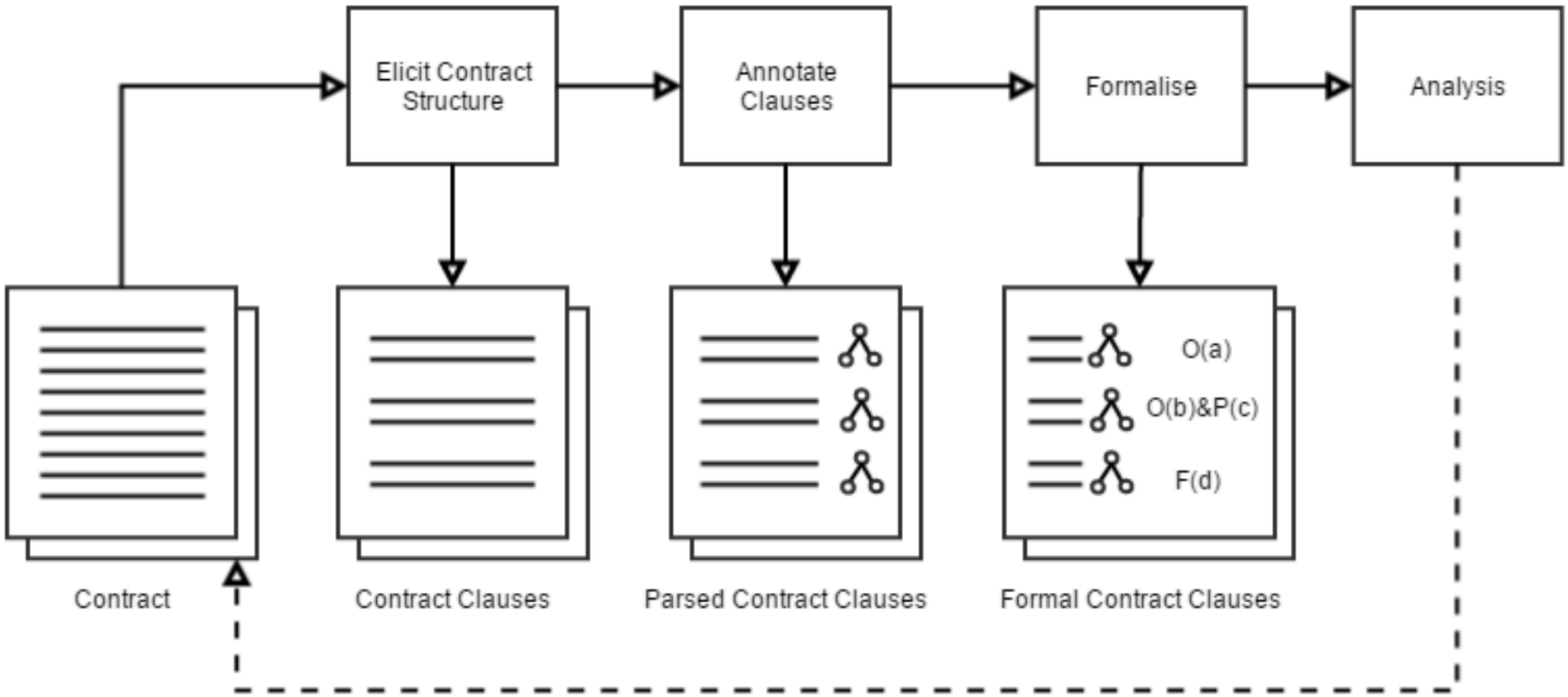
Natural Language Normative Texts



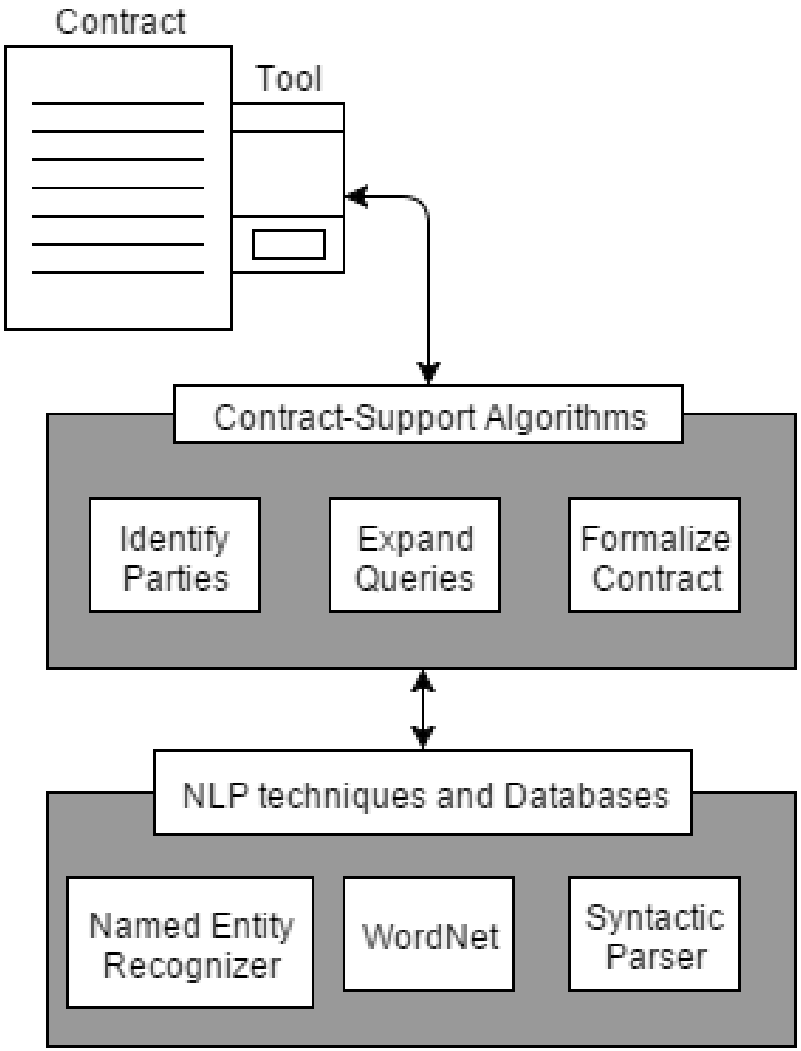
Natural Language Normative Texts



Natural Language Normative Texts



Natural Language Normative Texts



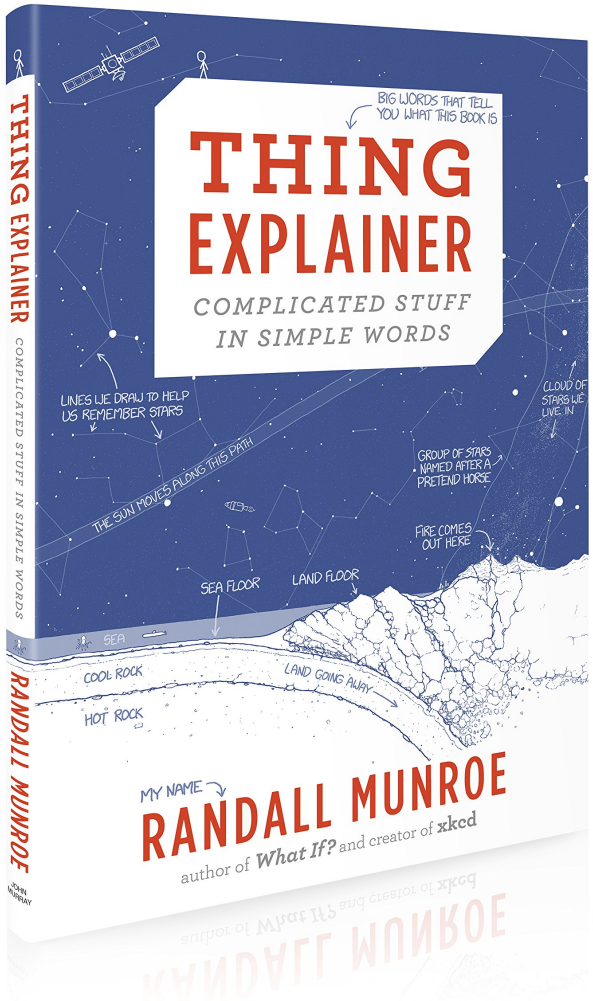
Some results

- The formal language we use limits what can be formalised (e.g. definition clauses cannot be formalised).
- The tool was evaluated based on how correct it was in formalising clauses which were in its domain of application.
- Evaluated against a random selection of contracts from the Australian Contract Corpus using hand-tagged clauses as the gold-standard.
- Looking at the analysis as a combination of (i) a classifier whether something can be formalised; and (ii) whether it was correctly formalised:
  - The tool managed to classify correctly 79% of the clauses.
  - Only 38% of the formalised clauses were correctly parsed – mostly due to non-normative clauses.
- Much work still needs to be done to filter definitions and other such clauses.

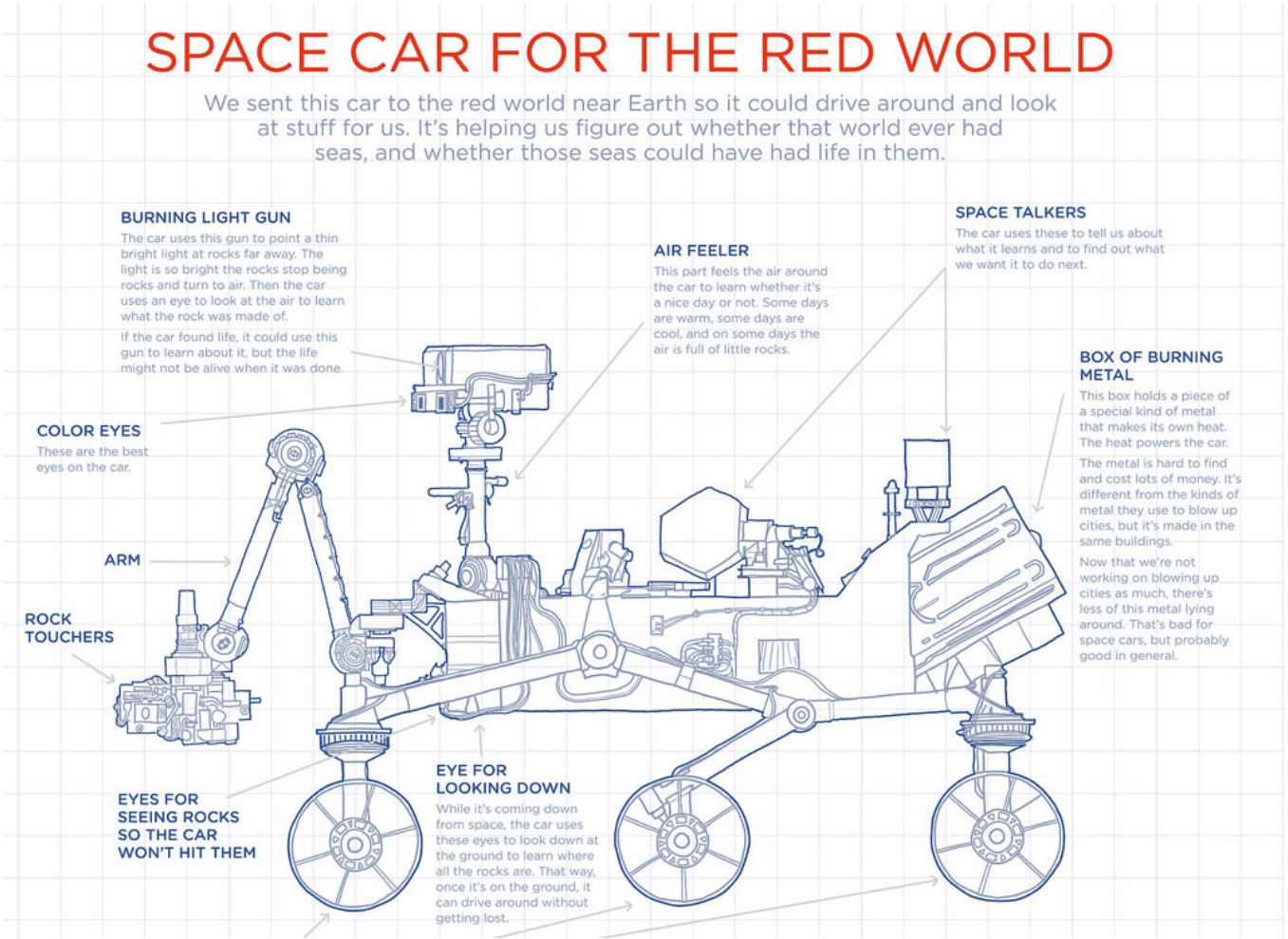
Controlled Natural Languages

- Controlled in terms of syntax.
- Controlled in terms of semantics – domain specific.
- Avoid ambiguity, enable analysis, generation, controlled expressiveness...
- and yet have a degree of naturalness.

Controlled Natural Languages: Human-Human



Controlled Natural Languages: Human-Human



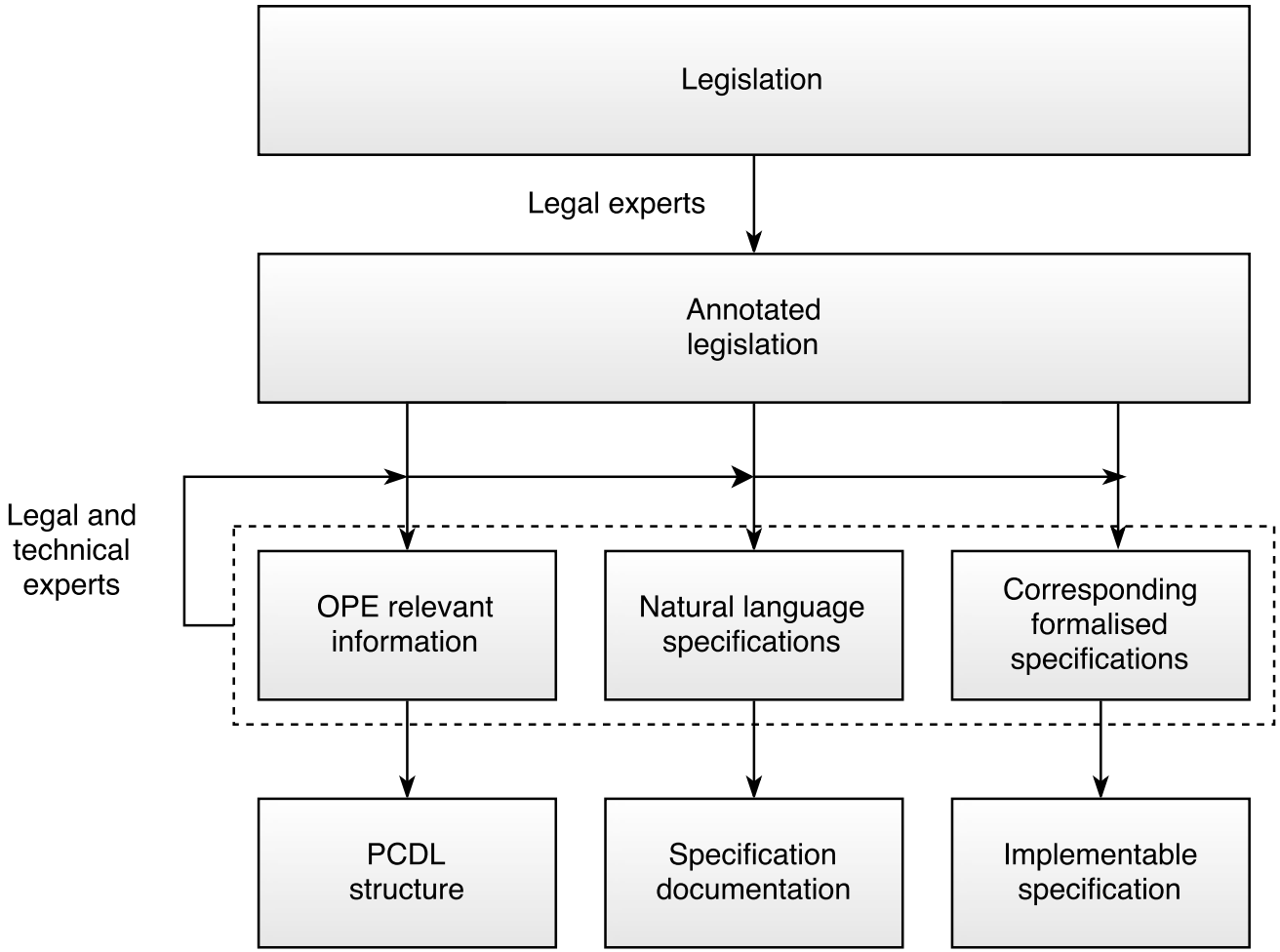
Controlled Natural Languages: Human-Machine/Machine-Human

"Load ID where for any 3 years, an individual declared average total chargeable income for the previous 3 years less than 2000 Euro."

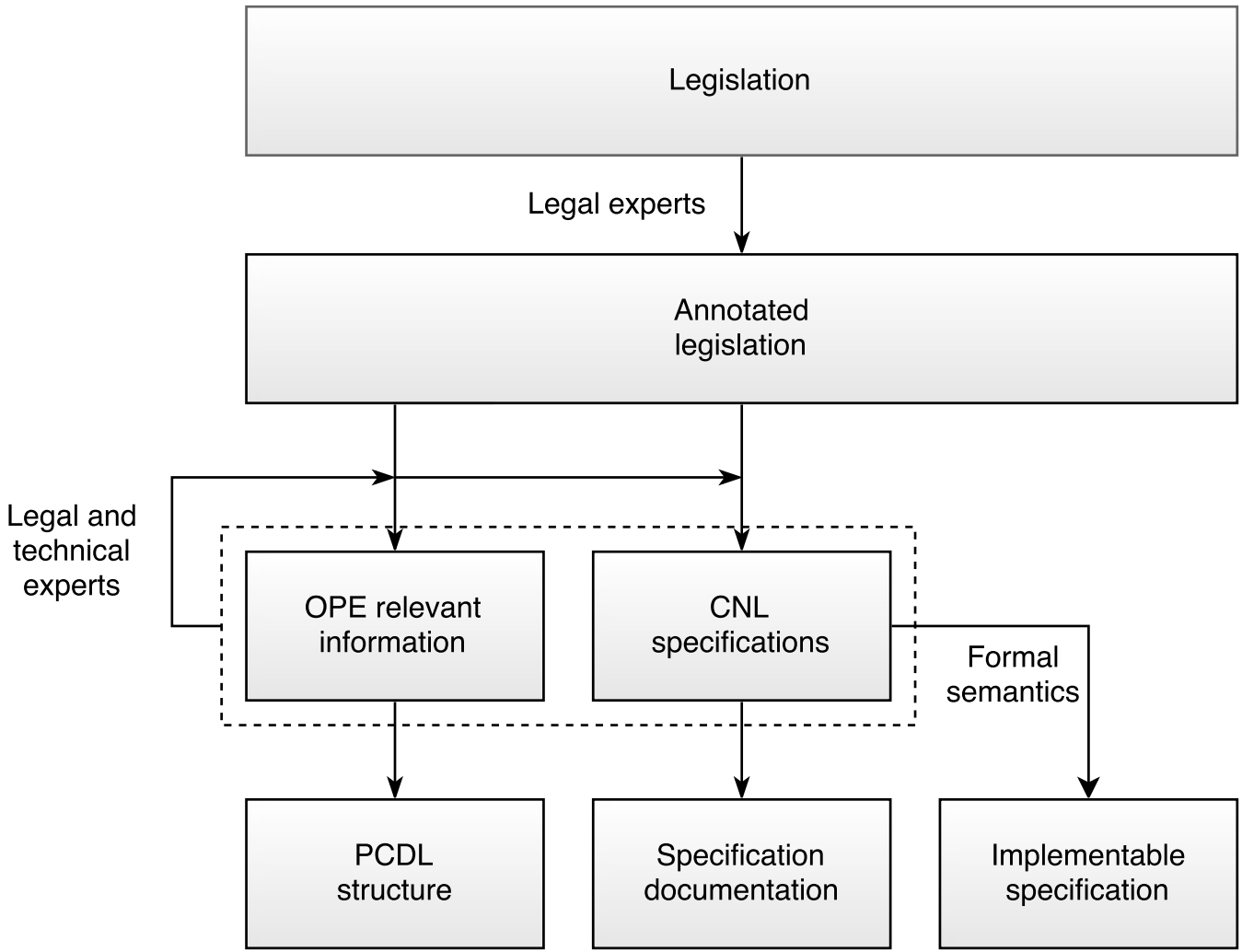
Controlled Natural Languages of Regulatory Texts

- Initial experiments with automated *Bananomic* (a variant of *Nomic*) game engine, implemented in GF:
  - *"If George is permitted to enact a rule then Paul is forbidden to abolish any rule."*
  - *"At some point before time 9 every player is obliged to throw a banana at George."*
- Other experiments with encoding rules regulating actions in an operation system, implemented in PC-PATR (and analysis in Haskell):
  - *"If SYSTEM accepts Job, then during one hour it is obligatory that SYSTEM make available results of Job unless SOMEONE cancels Job."*
  - *"It is permitted that only owner of Job cancels Job."*

Financial Service Regulations CNL



Financial Service Regulations CNL



Financial Service Regulations CNL

- H2020 project aimed at *"providing for the low-cost creation of quality assured payment applications and the building and delivery of payment services, in a controlled and regulatory-compliant environment."*
- Encoded all UK and Gibraltar legislation related to electronic financial transactions.
- Formalisation supported by a legal firm.
- Formalised regulations are used for static and dynamic analysis of payment applications.

Financial Service Regulations CNL

45. An electronic money issuer must not award–

(a) interest in respect of the holding of electronic money; or

(b) any other benefit related to the length of time during which an electronic money holder holds electronic money.

CC-3. For each programme p, and instrument i, where p is regulated in the UK, and i deals with e-money, then i does not give time-based rewards.

Formal semantics of normative texts

Formalising normative texts

- State vs. action based normative modalities.
- Beyond compliance – normative texts as first-class citizens.
- Challenges:
  - Temporal behaviour.
  - Deontic modalities, particularly permission.
  - Reparations.
  - Parties.

Deontic logics

$$
\begin{array}{rcl}
\mathcal{C} & ::= & \top \mid \bot \\
& \mid & \mathcal{P}_P(E) \mid \mathcal{O}_P(E) \mid \mathcal{F}_P(E) \\
& \mid & \mathcal{C} \wedge \mathcal{C} \mid \mathcal{C} + \mathcal{C} \mid \alpha.\mathcal{C} \\
& \mid & \mathcal{C}; \mathcal{C} \mid \mathcal{C} \triangleright \mathcal{C} \\
& \mid & \mu x.\mathcal{C} \mid x
\end{array}
$$

- Note temporal modalities appear both inside and outside deontic ones:

$$(\mathcal{O}_1(a); \mathcal{O}_1(b)) \triangleright C \neq \mathcal{O}_1(a; b) \triangleright C$$

- Negation inside and outside modalities may yield some equivalences such as:

$$\mathcal{O}_p(!e) = \mathcal{F}_p(e) = !\mathcal{P}_p(e)$$

Deontic logics

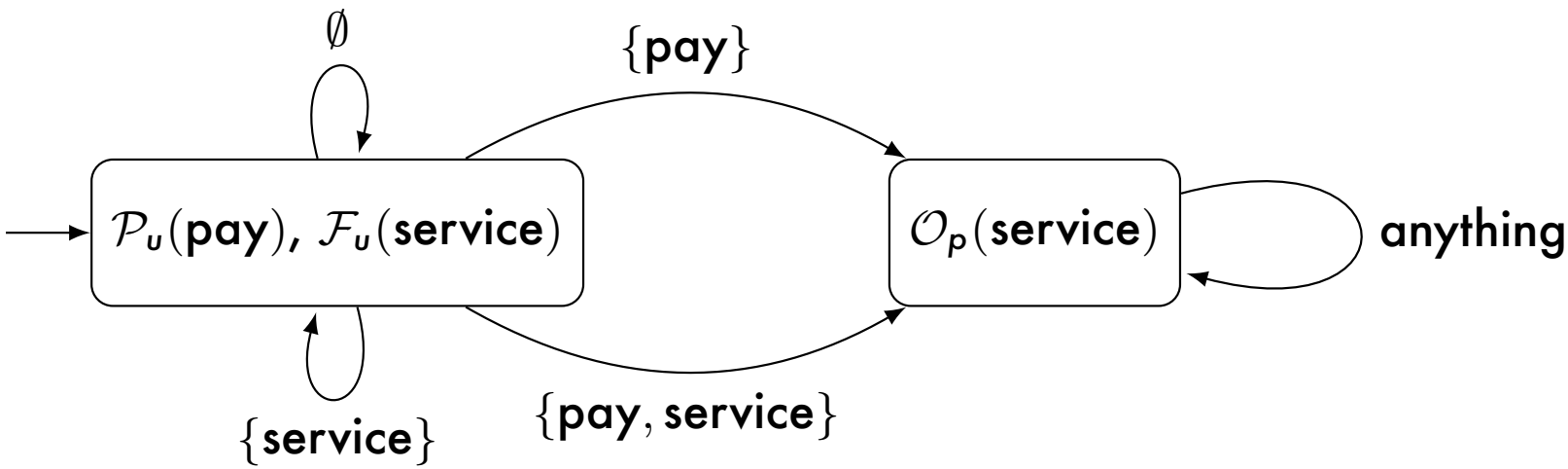
$$
\begin{array}{rcl}
\mathcal{C} & ::= & \top \mid \bot \\
& \mid & \mathcal{P}_P(E) \mid \mathcal{O}_P(E) \mid \mathcal{F}_P(E) \\
& \mid & \mathcal{C} \wedge \mathcal{C} \mid \mathcal{C} + \mathcal{C} \mid \alpha.\mathcal{C} \\
& \mid & \mathcal{C}; \mathcal{C} \mid \mathcal{C} \triangleright \mathcal{C} \\
& \mid & \mu x.\mathcal{C} \mid x
\end{array}
$$

- Trace-based semantics cannot deal with permission.
- Semantics either use sets of actions or use some form of persistence of deontic modalities.

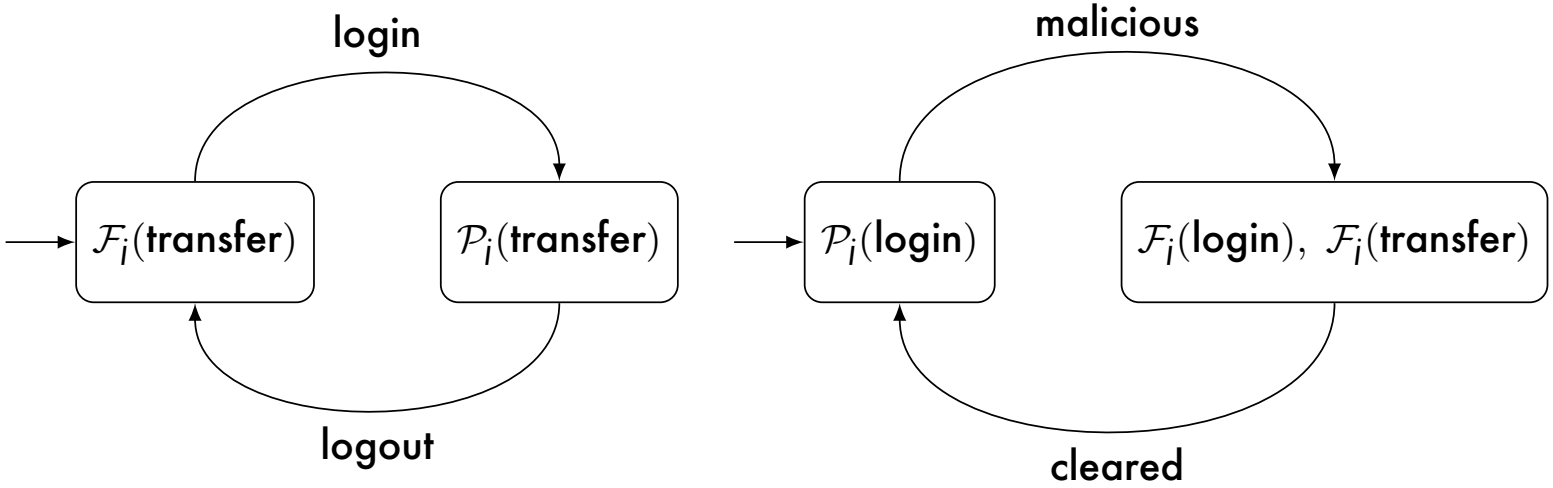
Automata-based formalisms

Contract

"Initially, the user (party u) is forbidden from using the service but permitted to pay after which the provider (party p) is obliged to provide the service."



Automata-based formalisms



- Conjunction = synchronous composition.
- Reparations semantically complex to deal with.

Partial contracts

$$
\begin{array}{rcl}
\hat{E} & ::= & ? \mid E \\
\hat{P} & ::= & ? \mid P \\
\hat{C} & ::= & ? \mid \top \mid \bot \\
& \mid & ?_{\hat{P}}(\hat{E}) \mid \mathcal{P}_{\hat{P}}(\hat{E}) \mid \mathcal{O}_{\hat{P}}(\hat{E}) \mid \mathcal{F}_{\hat{P}}(\hat{E}) \\
& \mid & \hat{C} \wedge \hat{C} \mid \hat{C} + \hat{C} \mid \alpha.\hat{C} \\
& \mid & \hat{C}; \hat{C} \mid \hat{C} \triangleright \hat{C} \\
& \mid & \mu x.\hat{C} \mid x
\end{array}
$$

- Especially when using NL techniques, parts of the text may be marked as *unparsed*.
- We can still analyse and compare texts using over- or under-approximations.
- Contrast with the use of three-valued logic in runtime verification.

Analysing normative formulae

Notions of equivalence

$C \equiv_p C'$ denotes C is equivalent to C' for party p

- Equivalence based on language of violations/satisfaction:

$$C \equiv_p C' \stackrel{df}{=} \mathcal{V}_p(C) = \mathcal{V}_p(C')$$

- Equivalence based on equivalence of judgement of violating systems:

$$C \equiv_p C' \stackrel{df}{=} \forall S_1,\ S_2 \ldots S_n \cdot \\ S_1 \| S_2 \| \ldots S_n \vdash_p C = S_1 \| S_2 \| \ldots S_n \vdash_p C$$

- Bisimulation-based approaches enable dealing with non-deterministic contracts and permission by introducing branching conditions in traces.

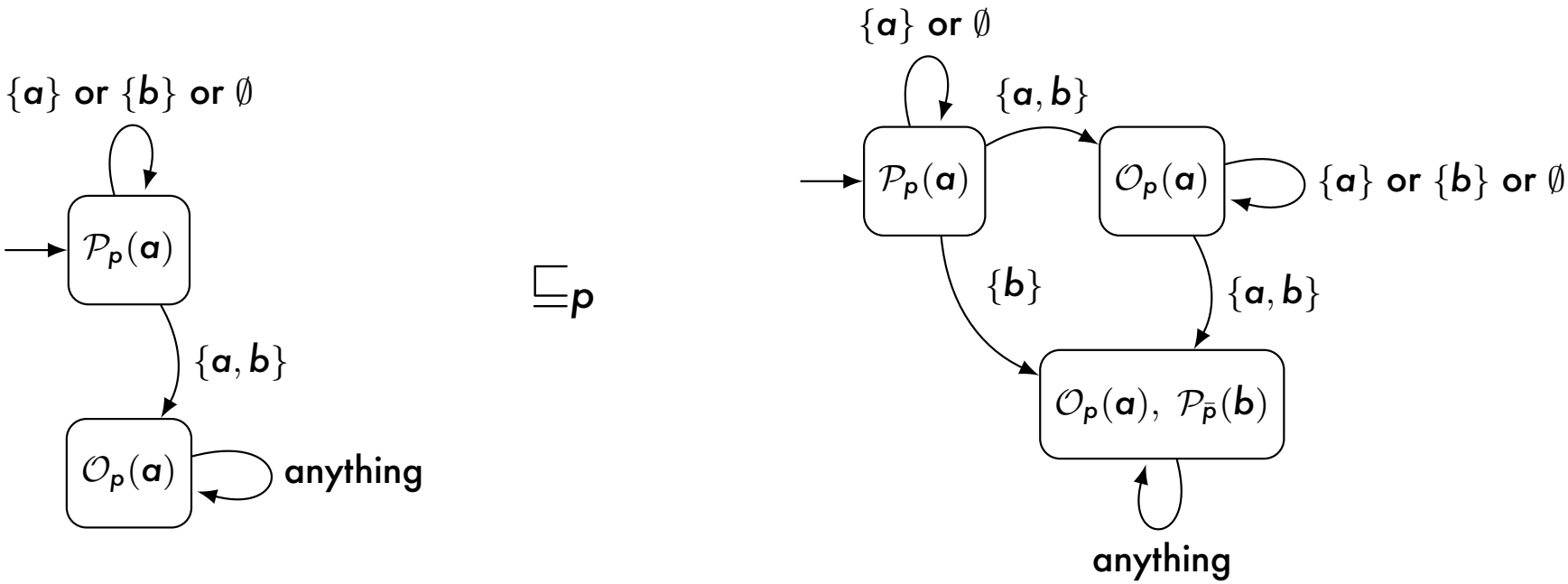
Strictness analysis

$C \sqsubseteq_p C'$ denotes that C' is stricter than C from the point-of-view of party p.

- Provides an ordering on contracts.
- Useful in contract negotiation.
- In the case of partially understood contracts, we have another partial order – the more we understand from a contract, the better it is.

Strictness analysis

$C \sqsubseteq_p C'$ denotes that C' is stricter than C from the point-of-view of party p.



Conflict detection

Axiom 1: Opposite permissions conflict:

$$\vdash \mathcal{P}_p(x) \maltese !\mathcal{P}_p(x)$$

Axiom 2: Obligation to perform mutually exclusive actions is a conflict:

$$a \bowtie b \vdash \mathcal{O}_p(a) \maltese \mathcal{O}_p(b)$$

Axiom 3: Conflicts are closed under symmetry:

$$C \maltese C' \vdash C' \maltese C$$

Axiom 4: Conflicts are closed under increased strictness:

$$C \maltese C' \wedge C' \sqsubseteq C'' \vdash C \maltese C''$$

Contracts and runtime monitoring, verification and enforcement

Runtime monitoring

- Runtime verification for legal compliance.
- Runtime verification for service contracts.
- Runtime enforcement for:
  - Legal obligations – e.g. *legal obligation to provide certain information to end users.*
  - Legal prohibitions – e.g. *legal prohibition of giving time-based rewards on e-money financial instruments.*
  - Legal permissions – e.g. *ensuring users have the option to redeem funds without fees.*

PrePost 2017: 19 September, in conjunction with iFM

Pre- and Post-Deployment Verification Techniques

Topics of interest

The workshop aims to bring together researchers working in the field of computer-aided validation, programming languages and verification to discuss the connections and interplay between pre- and post-deployment verification techniques.

- Monitoring, Enforcement and Adaptation
- Dynamic/Static/Gradual Type Systems
- Runtime Verification
- Model Checking
- Testing
- Program Logics

PrePost 2017: 19 September, in conjunction with iFM

Pre- and Post-Deployment Verification Techniques

Topics of interest

The workshop aims to bring together researchers working in the field of computer-a…ation to discuss the co…ment verification tech…

- Monit…
- Dyna…
- Runtir…
- Mode…
- Testin…
- Program Logics

Important Dates

Proceedings to be published in EPTCS, awaiting reply for special edition of JLAP with selected papers.

Abstract submission	June 5, 2017
Paper submission	June 12, 2017
Notification of acceptance	July 10, 2017
Camera ready version	July 17, 2017
Workshop PrePost 2017	Sep 19, 2017