

A Controlled Natural Language for Financial Services Compliance Checking¹

Shaun Azzopardi, Christian Colombo, Gordon J. Pace

Department of Computer Science, University of Malta

August 27, 2018

¹This research has received funding from the European Union's Horizon 2020 research and innovation programme under grant number 666363.

Introduction

- Financial services exist under high scrutiny and regulation
- But ensuring compliance is difficult, and especially for developers with no previous experience in the field

Introduction

- Financial services exist under high scrutiny and regulation
- But ensuring compliance is difficult, and especially for developers with no previous experience in the field
- In financial software we can automate compliance (to an extent).

Introduction

- Financial services exist under high scrutiny and regulation
- But ensuring compliance is difficult, and especially for developers with no previous experience in the field
- In financial software we can automate compliance (to an extent).
- But there is a gap between code and regulation text.

Introduction

- Financial services exist under high scrutiny and regulation
- But ensuring compliance is difficult, and especially for developers with no previous experience in the field
- In financial software we can automate compliance (to an extent).
- But there is a gap between code and regulation text.
- CNLs can bridge this gap, with appropriate semantics allowing generation of automated compliance checks

Introduction

- Financial services exist under high scrutiny and regulation
- But ensuring compliance is difficult, and especially for developers with no previous experience in the field
- In financial software we can automate compliance (to an extent).
- But there is a gap between code and regulation text.
- CNLs can bridge this gap, with appropriate semantics allowing generation of automated compliance checks
- We motivate and present the Financial Services Regulations Controlled Natural Language (**FSRCNL**) aimed at **specifying regulations for the purpose of verifying** that financial service applications satisfy them.

Introduction

- Financial services exist under high scrutiny and regulation
- But ensuring compliance is difficult, and especially for developers with no previous experience in the field
- In financial software we can automate compliance (to an extent).
- But there is a gap between code and regulation text.
- CNLs can bridge this gap, with appropriate semantics allowing generation of automated compliance checks
- We motivate and present the Financial Services Regulations Controlled Natural Language (**FSRCNL**) aimed at **specifying regulations for the purpose of verifying** that financial service applications satisfy them.
- This was done in the context of the Open Payments Ecosystem (OPE), an ecosystem for financial services applications

- ① The Regulations
 - ① Relevant and Verifiable Regulations
 - ② Features of the Relevant and Verifiable Subset
 - ③ Annotation and Formalisation Process
- ② The Language - FSRCNL
 - ① Semantics
 - ② Discussion - Design Choices
- ③ Conclusions

Relevant and Verifiable Regulations

- Not all clauses are relevant to our limited scope of the OPE:

Relevant and Verifiable Regulations

- Not all clauses are relevant to our limited scope of the OPE:
 - **PSR Schedule 1 1(d)** [...] transaction where the funds are covered by a credit line for the payment service user[...]

Relevant and Verifiable Regulations

- Not all clauses are relevant to our limited scope of the OPE:
 - **PSR Schedule 1 1(d)** [...] transaction where the funds are covered by a credit line for the payment service user[...]
- Not all clauses can be automatically checked:

Relevant and Verifiable Regulations

- Not all clauses are relevant to our limited scope of the OPE:
 - **PSR Schedule 1 1(d)** [...] transaction where the funds are covered by a credit line for the payment service user[...]
- Not all clauses can be automatically checked:
 - **PSR Schedule 4 40. (1)** A payment service provider must provide to the payment service user the information specified in Schedule 4[...]

Relevant and Verifiable Regulations

- Not all clauses are relevant to our limited scope of the OPE:
 - **PSR Schedule 1 1(d)** [...] transaction where the funds are covered by a credit line for the payment service user[...]
- Not all clauses can be automatically checked:
 - **PSR Schedule 4 40. (1)** A payment service provider must provide to the payment service user the information specified in Schedule 4[...]
- Then, we did not need to check for the whole regulations, but only for the *relevant* and *verifiable* clauses.

Relevant and Verifiable Regulations

Regulation Title	No.
The Electronic Money Regulations 2011 (SI 2011/99)	11
The Payment Services Regulations 2009 (SI 2009/209)	14
The Money Laundering Regulations 2009 (SI 2009/209)	4
Fourth Money Laundering Directive (EU) 2015/849	0
European Commissions Proposal for a Directive Amending MLD4	2

Relevant and Verifiable Examples: Definitions

- **EMR2(1)** electronic money means electronically (including magnetically) stored monetary value as represented by a claim on the electronic money issuer which
 - (a) is issued on receipt of funds for the purpose of making payment transactions;
[. . .]

Relevant and Verifiable Examples: Obligations and Prohibitions

- **EMR45** 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.

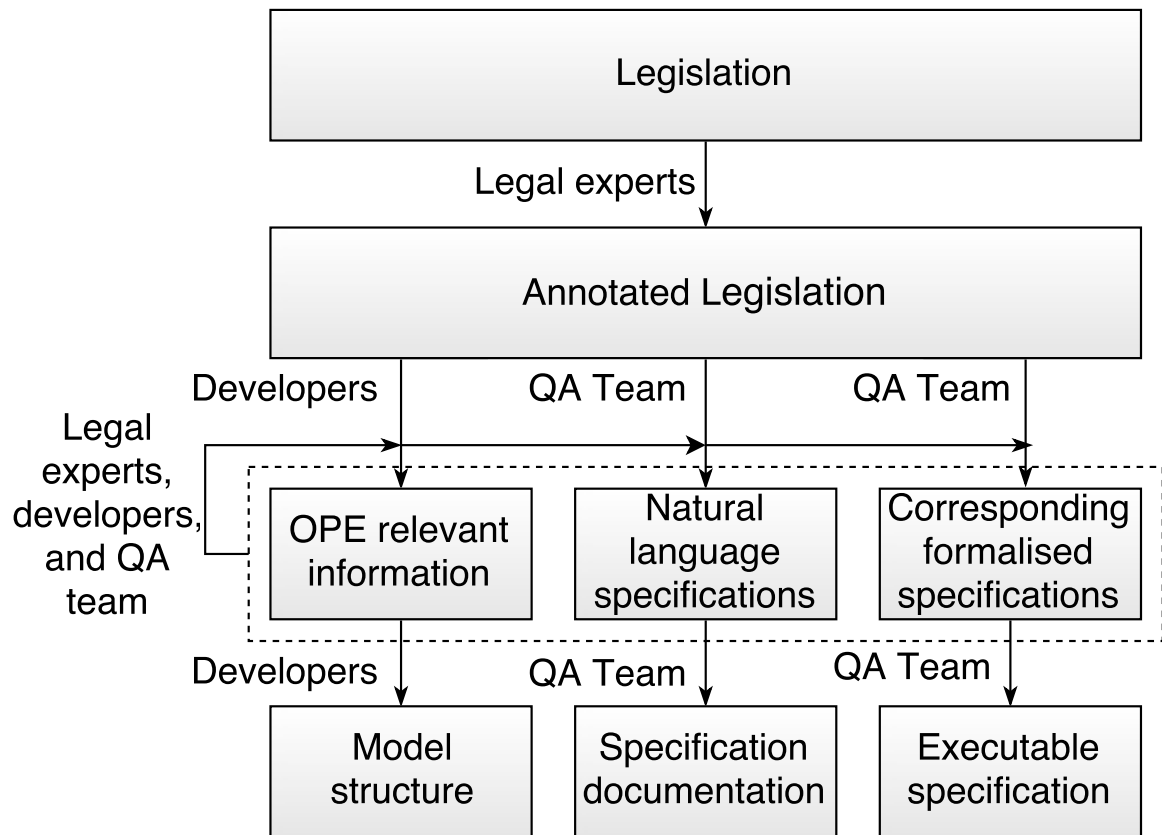
Relevant and Verifiable Examples: Monetary and Temporal Conditions

- **ML13(7)(d)(ii)** [. . .] if the device can be recharged, **a limit of 2,500 euro** is imposed on the total amount transacted in a calendar year, except when **an amount of 1,000 euro or more is redeemed in the same calendar year** by the bearer [. . .]

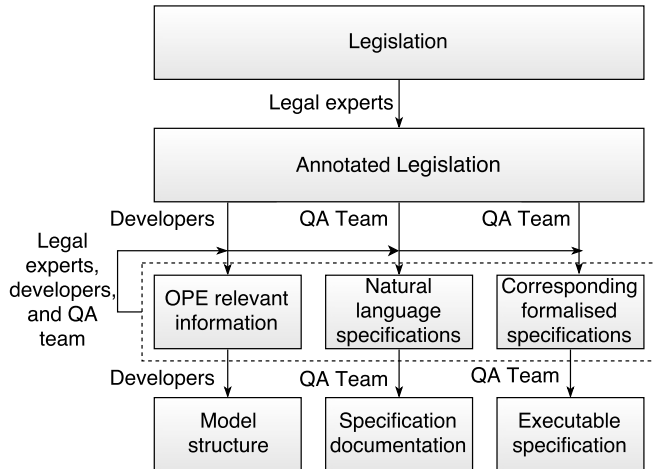
Features of the Relevant and Verifiable Regulation Subset

- They specify what **should (or should not)** take place, depending on some **constraint**
- They put **limits** on some **monetary** transactions
- Other obligations can trigger given some **time or monetary constraint**

Regulation Formalisation Process

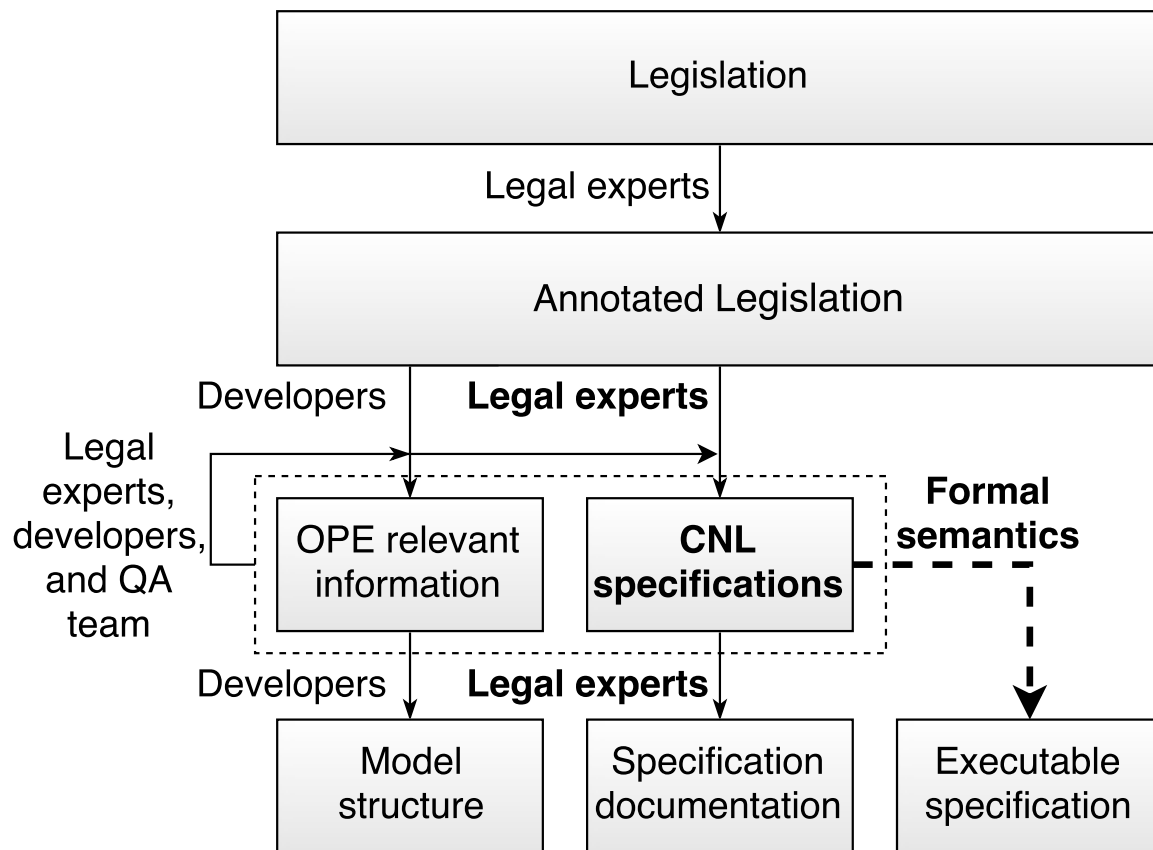


Regulation Formalisation Process



Issue: Manually producing and maintaining three sets of specifications is intensive, and leaves much room for inconsistency.

Solution: Regulation Formalisation Process with CNL



Financial Services Regulations CNL (FSRCNL)

- A controlled natural language (in English) geared towards specifying financial services regulations for compliance checking
- Development driven by the OPE project.

Financial Services Regulations CNL (FSRCNL)

- A controlled natural language (in English) geared towards specifying financial services regulations for compliance checking
- Development driven by the OPE project.
- Constituents:
 - **Variable declaration** (e.g. *programme p*, *transaction t*, or *instrument i*)

Financial Services Regulations CNL (FSRCNL)

- A controlled natural language (in English) geared towards specifying financial services regulations for compliance checking
- Development driven by the OPE project.
- Constituents:
 - **Variable declaration** (e.g. *programme p, transaction t, or instrument i*)
 - **Subject-Verb-Object phrases** (e.g. *i deals with e-money, i does not deal with e-money*)

Financial Services Regulations CNL (FSRCNL)

- A controlled natural language (in English) geared towards specifying financial services regulations for compliance checking
- Development driven by the OPE project.
- Constituents:
 - **Variable declaration** (e.g. *programme p*, *transaction t*, or *instrument i*)
 - **Subject-Verb-Object phrases** (e.g. *i deals with e-money*, *i does not deal with e-money*)
 - **Guarded declaration** (e.g. *service provider sp*, and *programme p*, where *sp deploys p in the UK*)

Financial Services Regulations CNL (FSRCNL)

- A controlled natural language (in English) geared towards specifying financial services regulations for compliance checking
- Development driven by the OPE project.
- Constituents:
 - **Variable declaration** (e.g. *programme p, transaction t, or instrument i*)
 - **Subject-Verb-Object phrases** (e.g. *i deals with e-money, i does not deal with e-money*)
 - **Guarded declaration** (e.g. *service provider sp, and programme p, where sp deploys p in the UK*)
 - **Monetary expressions** (e.g. *t deals with less than 500 EUR*)

Financial Services Regulations CNL (FSRCNL)

- A controlled natural language (in English) geared towards specifying financial services regulations for compliance checking
- Development driven by the OPE project.
- Constituents:
 - **Variable declaration** (e.g. *programme p, transaction t, or instrument i*)
 - **Subject-Verb-Object phrases** (e.g. *i deals with e-money, i does not deal with e-money*)
 - **Guarded declaration** (e.g. *service provider sp, and programme p, where sp deploys p in the UK*)
 - **Monetary expressions** (e.g. *t deals with less than 500 EUR*)
 - **Temporal and Country qualifiers** (e.g. *i expired less than 12 months ago, or p is regulated in the UK*)

Financial Services Regulations CNL (FSRCNL)

- A controlled natural language (in English) geared towards specifying financial services regulations for compliance checking
- Development driven by the OPE project.
- Constituents:
 - **Variable declaration** (e.g. *programme p*, *transaction t*, or *instrument i*)
 - **Subject-Verb-Object phrases** (e.g. *i deals with e-money*, *i does not deal with e-money*)
 - **Guarded declaration** (e.g. *service provider sp*, and *programme p*, where *sp deploys p in the UK*)
 - **Monetary expressions** (e.g. *t deals with less than 500 EUR*)
 - **Temporal and Country qualifiers** (e.g. *i expired less than 12 months ago*, or *p is regulated in the UK*)
 - **Full Sentence:** For each ⟨variable-declarations⟩, where ⟨guards⟩, then ⟨compound-sentence⟩.

- **ML13(7)(d)(ii)** [. . .] if the device can be recharged, a limit of 2,500 euro is imposed on the total amount transacted in a calendar year [. . .]
- *For each instrument i , where i is regulated in the UK and i is rechargeable, then the amount redeemed from i within a calendar year is exactly or less than 2500 EUR.*

- **EMR45** 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.

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

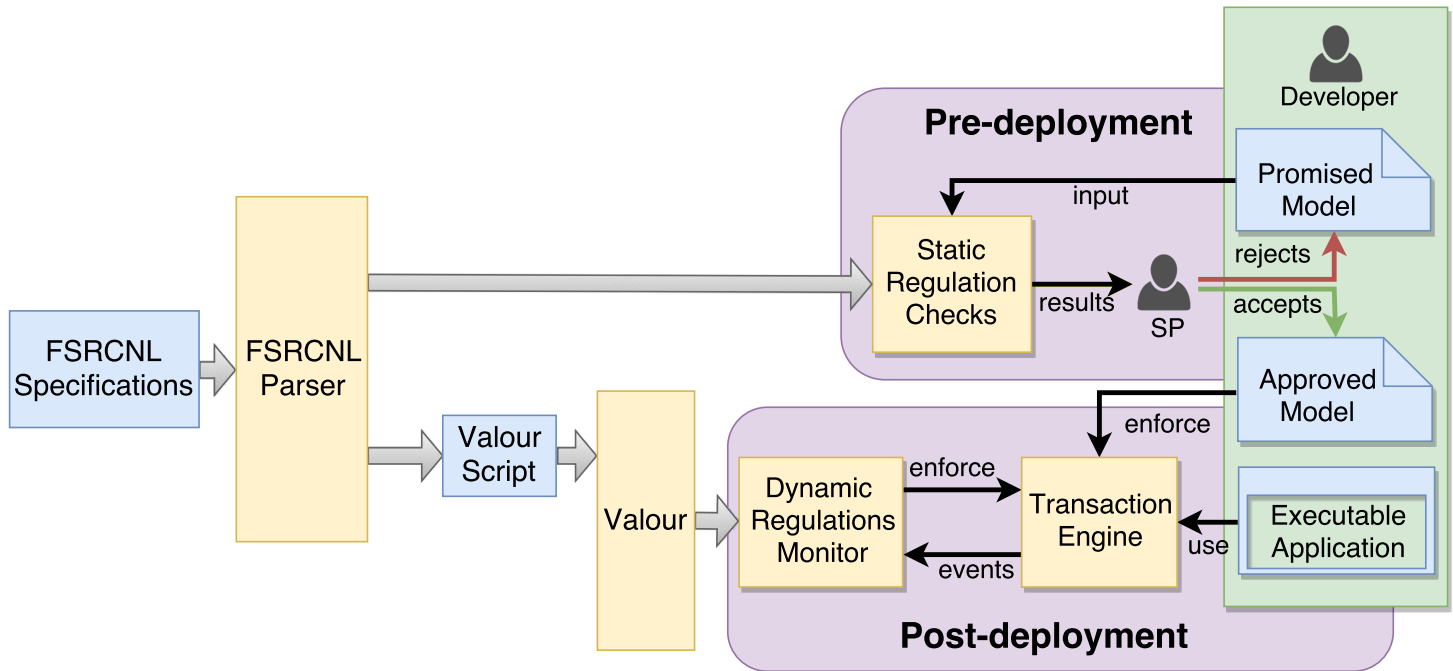
- FSRCNL sentences can be transformed into a predicate language:
 - *For each programme p , and instrument i , where i is an instrument of p , p is regulated in the UK, and i deals with e-money, then i does not give time-based rewards.*
 - $\forall p \in \text{programmes}, i \in \text{instruments}(p) \cdot \text{regulatedIn}(p, \text{UK}) \wedge \text{emoney}(i) \implies \text{noTimeRewards}(i)$

- FSRCNL sentences can be transformed into a predicate language:
 - *For each programme p , and instrument i , where i is an instrument of p , p is regulated in the UK, and i deals with e-money, then i does not give time-based rewards.*
 - $\forall p \in \text{programmes}, i \in \text{instruments}(p) \cdot \text{regulatedIn}(p, \text{UK}) \wedge \text{emoney}(i) \implies \text{noTimeRewards}(i)$
- Language constructs (that represent predicates) can be linked to constructs in a payment application.

Verification Context - Compliance in the Open Payments Ecosystem

- Open Payments Ecosystem (OPE) - An ecosystem acting as a backend for financial services, to be used by payment applications
- Developers provide
 - **code** for payment application, and a
 - **model** of the promised runtime behaviour (e.g. promising that only transactions between users in the UK will be allowed by the application)
- **FSRCNL types and verbs are linked either to the model, or to the code.**

OPE Business Process with Compliance



- We implemented a FSRCNL parser using Haskell's *parsec* library.

Discussion

- We implemented a FSRCNL parser using Haskell's *parsec* library.
- We use *parsec* to produce generic compliance checks (in Java) that can be connected to a system with an appropriate adaptor class.

- We implemented a FSRCNL parser using Haskell's *parsec* library.
- We use *parsec* to produce generic compliance checks (in Java) that can be connected to a system with an appropriate adaptor class.
- This work unfortunately lacks an extensive evaluation (due to project constraints), but briefly:
 - Developers found the FSRCNL regulations more straightforward
 - All identified regulations were able to be specified using FSRCNL.

- We implemented a FSRCNL parser using Haskell's *parsec* library.
- We use *parsec* to produce generic compliance checks (in Java) that can be connected to a system with an appropriate adaptor class.
- This work unfortunately lacks an extensive evaluation (due to project constraints), but briefly:
 - Developers found the FSRCNL regulations more straightforward
 - All identified regulations were able to be specified using FSRCNL.
 - FSRCNL was developed with two regulations (e-money and payment services) and tested for suitability with the money laundering regulations.
 - Only the addition of new verbs was needed.

- We implemented a FSRCNL parser using Haskell's *parsec* library.
- We use *parsec* to produce generic compliance checks (in Java) that can be connected to a system with an appropriate adaptor class.
- This work unfortunately lacks an extensive evaluation (due to project constraints), but briefly:
 - Developers found the FSRCNL regulations more straightforward
 - All identified regulations were able to be specified using FSRCNL.
 - FSRCNL was developed with two regulations (e-money and payment services) and tested for suitability with the money laundering regulations.
 - Only the addition of new verbs was needed.
 - Outstanding question: How easy is it for lawyers to write these specifications?

- We tried to use the same terms as used in regulations, rather than those used by the developers in the OPE.

Discussion

- We tried to use the same terms as used in regulations, rather than those used by the developers in the OPE.
- Variable declarations are unnatural, but we use them to remove the ambiguity of anaphora.

- We tried to use the same terms as used in regulations, rather than those used by the developers in the OPE.
- Variable declarations are unnatural, but we use them to remove the ambiguity of anaphora.
- Some concepts in the law were ambiguous,
 - e.g. **EMR 39(a)** *issuing of e-money should be done without delay*
 - For the OPE we decided on checking for an approximate amount of needed processing time, after which there is a delay.

- We tried to use the same terms as used in regulations, rather than those used by the developers in the OPE.
- Variable declarations are unnatural, but we use them to remove the ambiguity of anaphora.
- Some concepts in the law where ambiguous,
 - e.g. **EMR 39(a)** *issuing of e-money should be done without delay*
 - For the OPE we decided on checking for an approximate amount of needed processing time, after which there is a delay.
- PENS Classification
 - P^4 : Not maximally precise since the semantics depend on the underlying system.
 - E^3 : We do not include second-order universal quantification.
 - N^3 : Variable declarations and lack of flow between different regulations cause some unnaturality.
 - S^4 : We have documented FSRCNL in less than 10 pages.

Conclusions

- We have described the analysis of regulations for the purpose of verification.
- We presented a CNL, FSRCNL, for the specification of financial services regulations, that includes monetary and temporal expressions and financial services specific constructs.
- We showed how this CNL is integral to the compliance process of a payment applications' ecosystem, the OPE.
- As far as semantics is concerned, FSRCNL seamlessly incorporates two sub-languages: (i) a language translated to pre-deployment checks on a model provided by the developer; and (ii) a language translated to runtime monitors on the code.