Specification and Analysis of Electronic Contracts

Gerardo Schneider
gerardo@ifi.uio.no

PMA group
Department of Informatics,
University of Oslo

Joint work with
Cristian Prisacariu (cristi@ifi.uio.no) and Gordon Pace
gordon.pace@um.edu.mt

University of California at Berkeley
Berkeley – 06 May, 2008
Contracts and Informatics

1. Conventional contracts
   - Traditional commercial and judicial domain

2. “Programming by contract” or “Design by contract” (e.g., Eiffel)
   - Relation between pre- and post-conditions of routines, method calls, invariants, temporal dependencies, etc

3. In the context of web services
   - Service-Level Agreement, usually written in an XML-like language (e.g. WSLA)

4. Behavioral interfaces
   - Specify the sequence of interactions between different participants.
   - The allowed interactions are captured by legal (sets of) traces

5. Contractual protocols
   - To specify the interaction between communicating entities

6. “Deontic e-contracts”: representing Obligations, Permissions, Prohibitions, Power, etc
   - Inspired from a conventional contract
   - Written directly in a formal specification language

7. “Social contracts”: Multi-agent systems
Contracts

- “A contract is a binding agreement between two or more persons that is enforceable by law.” [Webster on-line]
“A contract is a binding agreement between two or more persons that is enforceable by law.” [Webster on-line]

This deed of Agreement is made between:
1. [name], from now on referred to as Provider and
2. the Client.

INTRODUCTION
3. The Provider is obliged to provide the Internet Services as stipulated in this Agreement.
4. DEFINITIONS
   a) Internet traffic may be measured by both Client and Provider by means of Equipment and may take the two values high and normal.

OPERATIVE PART
1. The Client shall not supply false information to the Client Relations Department of the Provider.
2. Whenever the Internet Traffic is high then the Client must pay [price] immediately, or the Client must notify the Provider by sending an e-mail specifying that he will pay later.
3. If the Client delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the normal level, and pay later twice (2 * [price]).
4. If the Client does not lower the Internet traffic immediately, then the Client will have to pay 3 * [price].
5. The Client shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the Provider’s web page to the Client Relations Department of the Provider.
Contracts

- We call the above a conventional contract
- An e-contract is a machine-readable contract

Two scenarios:

1. Obtain an e-contract from a conventional contract
   - Context: legal (e.g. financial) contracts
2. Write the e-contract directly in a formal language
   - Context: web services, components, OO, etc

Definition

A contract is a document which engages several parties in a transaction and stipulates their (conditional) obligations, rights, and prohibitions, as well as penalties in case of contract violations.

- A better name: ‘deontic’ e-contracts
Contracts

- We call the above a *conventional contract*
- An *e-contract* is a machine-readable contract

Two scenarios:

1. Obtain an e-contract from a conventional contract
   - Context: legal (e.g. financial) contracts
2. Write the e-contract directly in a formal language
   - Context: web services, components, OO, etc

**Definition**

A *contract* is a document which engages several parties in a transaction and stipulates their (conditional) *obligations, rights, and prohibitions*, as well as *penalties in case of contract violations*.

- A better name: ‘*deontic*’ e-contracts
Aim and Motivation

1. Use **deontic e-contracts** to ‘rule’ services exchange

2. Give a formal language for specifying/writing contracts

3. Analyze contracts “internally”
   - Detect contradictions/inconsistencies statically
   - Determine the obligations (permissions, prohibitions) of a signatory
   - Detect superfluous contract clauses

4. Develop a theory of contracts
   - Contract composition
   - Subcontracting
   - Conformance between a contract and the governing policies
   - *Meta-contracts* (policies)

5. Monitor contracts
   - Run-time system to ensure the contract is respected
   - In case of contract violations, act accordingly
Aim and Motivation

Use deontic e-contracts to ‘rule’ services exchange

1. Give a formal language for specifying/writing contracts
2. Analyze contracts “internally”
   - Detect contradictions/inconsistencies statically
   - Determine the obligations (permissions, prohibitions) of a signatory
   - Detect superfluous contract clauses
3. Develop a theory of contracts
   - Contract composition
   - Subcontracting
   - Conformance between a contract and the governing policies
   - Meta-contracts (policies)
4. Monitor contracts
   - Run-time system to ensure the contract is respected
   - In case of contract violations, act accordingly
Aim and Motivation

- Use **deontic e-contracts** to ‘rule’ services exchange

1. **Give a formal language** for specifying/writing contracts
2. **Analyze** contracts “internally”
   - Detect contradictions/inconsistencies statically
   - Determine the obligations (permissions, prohibitions) of a signatory
   - Detect superfluous contract clauses
3. **Develop a theory of contracts**
   - Contract composition
   - Subcontracting
   - Conformance between a contract and the governing policies
   - *Meta-contracts* (policies)
4. **Monitor contracts**
   - Run-time system to ensure the contract is respected
   - In case of contract violations, act accordingly
Aim and Motivation

- Use **deontic e-contracts** to ‘rule’ services exchange

1. **Give a formal language** for specifying/writing contracts
2. **Analyze** contracts “internally”
   - Detect contradictions/inconsistencies statically
   - Determine the obligations (permissions, prohibitions) of a signatory
   - Detect superfluous contract clauses
3. **Develop a theory of contracts**
   - Contract composition
   - Subcontracting
   - Conformance between a contract and the governing policies
   - *Meta-contracts* (policies)
4. **Monitor contracts**
   - Run-time system to ensure the contract is respected
   - In case of contract violations, act accordingly
Aim and Motivation

- Use **deontic e-contracts** to ‘rule’ services exchange

1. **Give a formal language** for specifying/writing contracts
2. **Analyze** contracts “internally”
   - Detect contradictions/inconsistencies statically
   - Determine the obligations (permissions, prohibitions) of a signatory
   - Detect superfluous contract clauses
3. **Develop a theory of contracts**
   - Contract composition
   - Subcontracting
   - Conformance between a contract and the governing policies
   - *Meta-contracts* (policies)
4. **Monitor** contracts
   - Run-time system to ensure the contract is respected
   - In case of contract violations, act accordingly
COSoDIS: “Contract-Oriented Software Development for Internet Services” – A Nordunet3 project (http://www.ifi.uio.no/cosodis/)
A Formal Language for Contracts

- A precise and concise syntax and a formal semantics
- Expressive enough as to capture natural contract clauses
- Restrictive enough to avoid (deontic) paradoxes and be amenable to formal analysis
  - Model checking
  - Deductive verification
- Allow representation of complex clauses: conditional obligations, permissions, and prohibitions
- Allow specification of (nested) contrary-to-duty (CTD) and contrary-to-prohibition (CTP)
  - CTD: when an obligation is not fulfilled
  - CTP: when a prohibition is violated
- We want to combine
  - The logical approach (e.g., dynamic, temporal, deontic logic)
  - The automata-like approach (labelled Kripke structures)
A Formal Language for Contracts

- A precise and concise syntax and a formal semantics
- Expressive enough as to capture natural contract clauses
- Restrictive enough to avoid (deontic) paradoxes and be amenable to formal analysis
  - Model checking
  - Deductive verification
- Allow representation of complex clauses: conditional obligations, permissions, and prohibitions
- Allow specification of (nested) contrary-to-duty (CTD) and contrary-to-prohibition (CTP)
  - CTD: when an obligation is not fulfilled
  - CTP: when a prohibition is violated
- We want to combine
  - The logical approach (e.g., dynamic, temporal, deontic logic)
  - The automata-like approach (labelled Kripke structures)
Concerned with moral and normative notions

- obligation, permission, prohibition, optionality, power, indifference, immunity, etc

Focus on

- The logical consistency of the above notions
- The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems

Difficult to avoid puzzles and paradoxes

- Logical paradoxes, where we can deduce contradictory actions
- “Practical oddities”, where we can get counterintuitive conclusions

Approaches

- ought-to-do: expressions consider names of actions
  - “The Internet Provider must send a password to the Client”
- ought-to-be: expressions consider state of affairs (results of actions)
  - “The average bandwidth must be more than 20kb/s”

We’ll only consider obligation, permission and prohibition over actions

Assertions define the “state of affairs”
Concerned with moral and normative notions
- obligation, permission, prohibition, optionality, power, indifference, immunity, etc.

Focus on
- The logical consistency of the above notions
- The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems

Difficult to avoid puzzles and paradoxes
- Logical paradoxes, where we can deduce contradictory actions
- “Practical oddities”, where we can get counterintuitive conclusions

Approaches
- ought-to-do: expressions consider names of actions
  - “The Internet Provider must send a password to the Client”
- ought-to-be: expressions consider state of affairs (results of actions)
  - “The average bandwidth must be more than 20kb/s”

We’ll only consider obligation, permission and prohibition over actions

Assertions define the “state of affairs”
Concerned with moral and normative notions

- obligation, permission, prohibition, optionality, power, indifference, immunity, etc

Focus on

- The logical consistency of the above notions
- The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems

Difficult to avoid puzzles and paradoxes

- Logical paradoxes, where we can deduce contradictory actions
- “Practical oddities”, where we can get counterintuitive conclusions

Approaches

- ought-to-do: expressions consider names of actions
  - “The Internet Provider must send a password to the Client”
- ought-to-be: expressions consider state of affairs (results of actions)
  - “The average bandwidth must be more than 20kb/s”

We’ll only consider obligation, permission and prohibition over actions

Assertions define the “state of affairs”
(Standard) Deontic Logic

Few Words

- Concerned with moral and normative notions
  - obligation, permission, prohibition, optionality, power, indifference, immunity, etc

- Focus on
  - The logical consistency of the above notions
  - The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems

- Difficult to avoid puzzles and paradoxes
  - Logical paradoxes, where we can deduce contradictory actions
  - “Practical oddities”, where we can get counterintuitive conclusions

- Approaches
  - ought-to-do: expressions consider names of actions
    - “The Internet Provider must send a password to the Client”
  - ought-to-be: expressions consider state of affairs (results of actions)
    - “The average bandwidth must be more than 20kb/s”

- We’ll only consider obligation, permission and prohibition over actions

- Assertions define the “state of affairs”
Concerned with moral and normative notions
- *obligation*, *permission*, *prohibition*, *optionality*, *power*, *indifference*, *immunity*, etc

Focus on
- The logical consistency of the above notions
- The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems

Difficult to avoid *puzzles* and *paradoxes*
- Logical paradoxes, where we can deduce contradictory actions
- “Practical oddities”, where we can get counterintuitive conclusions

Approaches
- **ought-to-do**: expressions consider *names of actions*
  - “The Internet Provider *must send* a password to the Client”
- **ought-to-be**: expressions consider *state of affairs* (results of actions)
  - “The average bandwidth *must be* more than 20kb/s”

We’ll only consider *obligation*, *permission* and *prohibition* over *actions*

Assertions define the “state of affairs”
1. The Contract Language $\mathcal{CL}$

2. Properties of the Language

3. Verification of Contracts

4. Final Remarks
1. The Contract Language $\mathcal{CL}$

2. Properties of the Language

3. Verification of Contracts

4. Final Remarks
The Contract Specification Language $\mathcal{CL}$

\[\text{Contract} \; := \; D \; ; \; C\]
\[C \; := \; C_O \; | \; C_P \; | \; C_F \; | \; C \land C \; | \; [\alpha]C \; | \; \langle \alpha \rangle C \; | \; C \cup C \; | \; \bigcirc C \; | \; \square C\]
\[C_O \; := \; O(\alpha) \; | \; C_O \oplus C_O\]
\[C_P \; := \; P(\alpha) \; | \; C_P \oplus C_P\]
\[C_F \; := \; F(\alpha) \; | \; C_F \lor [\alpha]C_F\]

- $O(\alpha)$, $P(\alpha)$, $F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- $\alpha$ are actions given in the definition part $D$
  - +: choice
  - .: concatenation (sequencing)
  - &: concurrency
  - $\phi?$: test
- $\land$, $\lor$, and $\oplus$ are conjunction, disjunction, and exclusive disjunction
- $[\alpha]$ and $\langle \alpha \rangle$ are the action parameterized modalities of dynamic logic
- $\cup$, $\bigcirc$, and $\square$ correspond to temporal logic operators
The Contract Specification Language $\mathcal{CL}$

$$\text{Contract} \ := \ D \ ; \ C$$

$$C \ := \ C_O \ | \ C_P \ | \ C_F \ | \ C \ \land \ C \ | \ [\alpha]C \ | \ \langle \alpha \rangle C \ | \ C \ U \ C \ | \ \bigcirc C \ | \ \blacksquare C$$

$$C_O \ := \ O(\alpha) \ | \ C_O \ \oplus \ C_O$$

$$C_P \ := \ P(\alpha) \ | \ C_P \ \oplus \ C_P$$

$$C_F \ := \ F(\alpha) \ | \ C_F \ \lor \ [\alpha]C_F$$

- $O(\alpha), P(\alpha), F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- $\alpha$ are actions given in the definition part $\mathcal{D}$
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrency
  - $\phi?$ test
- $\land, \lor, \text{and } \oplus$ are conjunction, disjunction, and exclusive disjunction
- $[\alpha]$ and $\langle \alpha \rangle$ are the action parameterized modalities of dynamic logic
- $U$, $\bigcirc$, and $\blacksquare$ correspond to temporal logic operators
The Contract Specification Language $\mathcal{CL}$

$$Contract := \mathcal{D} ; \mathcal{C}$$

$$\mathcal{C} := \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \land \mathcal{C} \mid [\alpha] \mathcal{C} \mid \langle \alpha \rangle \mathcal{C} \mid \mathcal{C} \lor \mathcal{C} \mid \Box \mathcal{C}$$

- $O(\alpha), P(\alpha), F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions.
- $\alpha$ are actions given in the definition part $\mathcal{D}$.
- $\lor, \land, \land, \land$ are conjunction, disjunction, and exclusive disjunction.
- $[\alpha]$ and $\langle \alpha \rangle$ are the action parameterized modalities of dynamic logic.
- $\lor, \land, \land$ correspond to temporal logic operators.
The Contract Specification Language \( \mathcal{CL} \)

\[
\text{Contract} \quad := \quad D \; ; \; C \\
C \quad := \quad C_O \; \mid \; C_P \; \mid \; C_F \; \mid \; C \land C \; \mid \; [\alpha]C \; \mid \; (\alpha)C \; \mid \; C U C \; \mid \; C \triangleright C \; \mid \; C \blacklozenge C \\
C_O \quad := \quad O(\alpha) \; \mid \; C_O \oplus C_O \\
C_P \quad := \quad P(\alpha) \; \mid \; C_P \oplus C_P \\
C_F \quad := \quad F(\alpha) \; \mid \; C_F \lor [\alpha]C_F \\
\]

- \( O(\alpha) \), \( P(\alpha) \), \( F(\alpha) \) specify obligation, permission (rights), and prohibition (forbidden) over actions
- \( \alpha \) are actions given in the \text{definition} part \( D \)
  - + choice
  - \cdot concatenation (sequencing)
  - \& concurrency
  - \( \phi? \) test
- \( \land, \lor, \text{ and } \oplus \) are conjunction, disjunction, and exclusive disjunction
- \( [\alpha] \) and \( (\alpha) \) are the \text{action parameterized modalities} of dynamic logic
- \( \mathcal{U}, \bigcirc \), and \( \bigboxdot \) correspond to \text{temporal logic operators}
The Contract Specification Language $\mathcal{CL}$

\[
\text{Contract} \quad ::= \quad \mathcal{D} \; ; \; \mathcal{C} \\
\mathcal{C} \quad ::= \quad \mathcal{C}_O \; | \; \mathcal{C}_P \; | \; \mathcal{C}_F \; | \; \mathcal{C} \wedge \mathcal{C} \; | \; [\alpha] \mathcal{C} \; | \; \langle \alpha \rangle \mathcal{C} \; | \; \mathcal{C} \cup \mathcal{C} \; | \; \bigcirc \mathcal{C} \; | \; \Box \mathcal{C}
\]

\[
\mathcal{C}_O \quad ::= \quad \mathcal{O}(\alpha) \; | \; \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P \quad ::= \quad \mathcal{P}(\alpha) \; | \; \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F \quad ::= \quad \mathcal{F}(\alpha) \; | \; \mathcal{C}_F \lor [\alpha] \mathcal{C}_F
\]

- $\mathcal{O}(\alpha)$, $\mathcal{P}(\alpha)$, and $\mathcal{F}(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- $\alpha$ are actions given in the definition part $\mathcal{D}$
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrency
  - $\phi?$ test
- $\wedge$, $\lor$, and $\oplus$ are conjunction, disjunction, and exclusive disjunction
- $[\alpha]$ and $\langle \alpha \rangle$ are the action parameterized modalities of dynamic logic
- $\mathcal{U}$, $\bigcirc$, and $\Box$ correspond to temporal logic operators
Tests as actions: $\phi$?

- The behaviour of a test is like a guard; e.g. $\phi? \cdot a$ if the test succeeds then action $a$ is performed
- Tests are used to model conditions: $[\phi?]C$ is the same as $\phi \Rightarrow C$

Action negation $\overline{\alpha}$

- It represents all immediate traces that take us outside the trace of $\alpha$
- Involves the use of a canonic form of actions
- E.g.: consider two atomic actions $a$ and $b$ then $\overline{a \cdot b}$ is $b + a \cdot a$
Actions
Test and Negation

- **Tests** as actions: $\phi$?
  - The behaviour of a test is like a *guard*; e.g. $\varphi? \cdot a$ *if the test succeeds then action $a$ is performed*
  - Tests are used to model conditions: $[\varphi?]{C}$ is the same as $\varphi \Rightarrow C$

- **Action negation** $\overline{\alpha}$
  - It represents all immediate traces that take us outside the trace of $\alpha$
  - Involves the use of a *canonic form* of actions
  - E.g.: consider two atomic actions $a$ and $b$ then $a \cdot b$ is $b + a \cdot a$
Actions
Concurrent actions

- \(a \& b\)

- “The client must pay immediately, or the client must notify the service provider by sending an e-mail specifying that he delays the payment”

\[O(p) \oplus O(d \& n)\]

- \(O(d \& n) \equiv O(d) \land O(n)\)

- Action algebra enriched with a conflict relation to represent incompatible actions
  - \(a = \) “increase Internet traffic” and \(b = \) “decrease Internet traffic”
    - \(a \not\equiv_c b\)
    - \(O(a) \land O(b)\) gives an inconsistency
Actions

Concurrent actions

- $a \& b$
- “The client must pay immediately, or the client must notify the service provider by sending an e-mail specifying that he delays the payment”

\[ O(p) \oplus O(d \& n) \]

- $O(d \& n) \equiv O(d) \land O(n)$

Action algebra enriched with a conflict relation to represent incompatible actions

- $a =$ “increase Internet traffic” and $b =$ “decrease Internet traffic”
  - $a \neq_C b$
  - $O(a) \land O(b)$ gives an inconsistency
Expressing contrary-to-duty (CTD)

\[ O_C(\alpha) = O(\alpha) \land [\bar{\alpha}]C \]
More on the Contract Language
CTD and CTP

- Expressing **contrary-to-duty** (CTD)
  
  \[ O_C(\alpha) = O(\alpha) \land [\alpha]C \]

- Expressing **contrary-to-prohibition** (CTP)
  
  \[ F_C(\alpha) = F(\alpha) \land [\alpha]C \]
More on the Contract Language
CTD and CTP

- Expressing contrary-to-duty (CTD)
  \[ O_C(\alpha) = O(\alpha) \land [\bar{\alpha}]C \]

- Expressing contrary-to-prohibition (CTP)
  \[ F_C(\alpha) = F(\alpha) \land [\alpha]C \]

Example
“[...] the client must immediately lower the Internet traffic to the low level, and pay later twice. If the client does not lower the Internet traffic immediately, then the client will have to pay three times the price”

In \( CL \):  
\[ \Box (O_C(I) \land [I]\Diamond (O(p&p))) \]

where \( C = \Diamond O(p&p&p&p) \)
Translation into a variant of $\mu$-calculus ($\mathcal{C}\mu$)

The syntax of the $\mathcal{C}\mu$ logic

$$\varphi ::= P \mid Z \mid P_c \mid T \mid \neg \varphi \mid \varphi \land \varphi \mid [\gamma] \varphi \mid \mu Z.\varphi(Z)$$

Main differences with respect to the classical $\mu$-calculus:

1. $P_c$ is set of propositional constants $O_a$ and $F_a$, one for each basic action $a$

2. Multisets of basic actions: i.e. $\gamma = \{a, a, b\}$ is a label
Translation into a variant of \( \mu \)-calculus (\( C\mu \))

The syntax of the \( C\mu \) logic

\[ \varphi ::= P \mid Z \mid P_c \mid \top \mid \neg \varphi \mid \varphi \land \varphi \mid [\gamma] \varphi \mid \mu Z. \varphi(Z) \]

Main differences with respect to the classical \( \mu \)-calculus:

1. \( P_c \) is set of propositional constants \( O_a \) and \( F_a \), one for each basic action \( a \)

2. Multisets of basic actions: i.e. \( \gamma = \{a, a, b\} \) is a label
Obligation

\[ f^T (O(a \& b)) = \langle \{a, b\}\rangle (O_a \land O_b) \]
Example: Obligation

\[ f^T(O(a \& b)) = \langle \{a, b\} \rangle (O_a \land O_b) \]
Very recent work:

- "The" semantics of the language
  - Branching semantics

- For monitoring purposes
  - Trace semantics
Outline

1. The Contract Language $\mathcal{CL}$
2. Properties of the Language
3. Verification of Contracts
4. Final Remarks
Ross’s paradox

1. It is obligatory that one mails the letter
2. It is obligatory that one mails the letter or one destroys the letter

In Standard Deontic Logic (SDL) these are expressed as:

1. $O(p)$
2. $O(p \lor q)$

Problem: in SDL one can infer that $O(p) \Rightarrow O(p \lor q)$

Avoided in $\mathcal{CL}$ – Proof Sketch:

- $f^T(O(a)) = \langle a \rangle O_a$
- $O(a + b) \equiv O(a) \oplus O(b) = \langle a \rangle O_a \land \langle b \rangle O_b$
- $\langle a \rangle O_a \not\Rightarrow \langle a \rangle O_a \land \langle b \rangle O_b$
Ross’s paradox

1. It is obligatory that one mails the letter
2. It is obligatory that one mails the letter or one destroys the letter

In Standard Deontic Logic (SDL) these are expressed as:

1. $O(p)$
2. $O(p \lor q)$

Problem: in SDL one can infer that $O(p) \Rightarrow O(p \lor q)$

Avoided in $\mathcal{CL}$ – Proof Sketch:

- $f^T(O(a)) = \langle a \rangle O_a$
- $O(a + b) \equiv O(a) \oplus O(b) \overset{f^T}{\equiv} \langle a \rangle O_a \land \langle b \rangle O_b$
- $\langle a \rangle O_a \not\Rightarrow \langle a \rangle O_a \land \langle b \rangle O_b$
The following paradoxes are avoided in CL:

- Ross’s paradox
- The Free Choice Permission paradox
- Sartre’s dilemma
- The Good Samaritan paradox
- Chisholm’s paradox
- The Gentle Murderer paradox
Theorem

The following hold in $\mathcal{CL}$:

- $P(\alpha) \equiv \neg F(\alpha)$
- $O(\alpha) \Rightarrow P(\alpha)$
- $P(a) \not\Rightarrow P(a \& b)$
- $F(a) \not\Rightarrow F(a \& b)$
- $F(a \& b) \not\Rightarrow F(a)$
- $P(a \& b) \not\Rightarrow P(a)$
1. The Contract Language $CL$

2. Properties of the Language

3. Verification of Contracts

4. Final Remarks
A **model checker** is a software tool that given:

- A model $M$ (usually a *Kripke model*)
- A property $\phi$ (usually a *temporal logic formula*)

It decides whether

$$M \models \phi$$

- It returns YES if the property is satisfied,
- Otherwise returns NO and provides a counterexample

It is completely automatic!
Model Checking Contracts

1. Model the conventional contract (in English) as a CL expression
2. Translate the CL specification into Cµ
3. Obtain a Kripke-like model (LTS) from the Cµ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   - Check the model is ‘good’
   - Check some properties about the client and the provider
6. In case of a counter-example given by NuSMV, interpret it as a CL clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract
Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}_\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}_\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   - Check the model is ‘good’
   - Check some properties about the client and the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract
1. Model the conventional contract (in English) as a $CL$ expression
2. Translate the $CL$ specification into $C\mu$
3. Obtain a Kripke-like model (LTS) from the $C\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   - Check the model is ‘good’
   - Check some properties about the client and the provider
6. In case of a counter-example given by NuSMV, interpret it as a $CL$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract
Model Checking Contracts

1. Model the conventional contract (in English) as a $CL$ expression
2. Translate the $CL$ specification into $C\mu$
3. Obtain a Kripke-like model (LTS) from the $C\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   - Check the model is ‘good’
   - Check some properties about the client and the provider
6. In case of a counter-example given by NuSMV, interpret it as a $CL$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract
Model Checking Contracts

1. Model the conventional contract (in English) as a \( CL \) expression
2. Translate the \( CL \) specification into \( C\mu \)
3. Obtain a Kripke-like model (LTS) from the \( C\mu \) formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   - Check the model is ‘good’
   - Check some properties about the client and the provider
6. In case of a counter-example given by NuSMV, interpret it as a \( CL \) clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract
1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}_\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}_\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   - Check the model is ‘good’
   - Check some properties about the client and the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract
Model Checking Contracts

1. Model the conventional contract (in English) as a $CL$ expression
2. Translate the $CL$ specification into $C\mu$
3. Obtain a Kripke-like model (LTS) from the $C\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   - Check the model is ‘good’
   - Check some properties about the client and the provider
6. In case of a counter-example given by NuSMV, interpret it as a $CL$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract
1. The **Client** shall not:
   a) supply false information to the Client Relations Department of the **Provider**.
2. Whenever the Internet Traffic is **high** then the **Client** must pay \([price]\) immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice \((2 \times [price])\).
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay \(3 \times [price]\).
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**’s web page to the Client Relations Department of the **Provider**.
6. **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
   a) Suspend Internet Services immediately if **Client** is in breach of Clause 1;
1. The **Client** shall not:
   a) supply false information to the Client Relations Department of the **Provider**.
2. Whenever the Internet Traffic is **high** then the **Client** must pay \([price]\) immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice \((2 \times [price])\).
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay \(3 \times [price]\).
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**’s web page to the Client Relations Department of the **Provider**.
6. **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
   a) Suspend Internet Services immediately if **Client** is in breach of Clause 1;
1. $\square F(fi)$

2. Whenever the Internet Traffic is high then the Client must pay $[price]$ immediately, or the Client must notify the Provider by sending an e-mail specifying that he will pay later.

3. If the Client delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the normal level, and pay later twice ($2 \times [price]$).

4. If the Client does not lower the Internet traffic immediately, then the Client will have to pay $3 \times [price]$.

5. The Client shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the Provider’s web page to the Client Relations Department of the Provider.

6. Provider may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
   a) Suspend Internet Services immediately if Client is in breach of Clause 1;
1. $\Box F(f_i)$

2. Whenever the Internet Traffic is high then the Client must pay $[price]$ immediately, or the Client must notify the Provider by sending an e-mail specifying that he will pay later.

3. If the Client delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the normal level, and pay later twice $(2 \times [price])$.

4. If the Client does not lower the Internet traffic immediately, then the Client will have to pay $3 \times [price]$.

5. The Client shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the Provider’s web page to the Client Relations Department of the Provider.

6. Provider may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
   a) Suspend Internet Services immediately if Client is in breach of Clause 1;
1. $\square F_{P(s)}(fi)$

2. Whenever the Internet Traffic is high then the Client must pay [price] immediately, or the Client must notify the Provider by sending an e-mail specifying that he will pay later.
3. If the Client delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the normal level, and pay later twice ($2 \times [price]$).
4. If the Client does not lower the Internet traffic immediately, then the Client will have to pay $3 \times [price]$.
5. The Client shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the Provider’s web page to the Client Relations Department of the Provider.
1. $\Box F_{P(s)}(fi)$

2. $\Box [h](\phi \Rightarrow O(p + (d \& n)))$

3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice $(2 \times [price])$.

4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 \times [price]$.

5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**’s web page to the Client Relations Department of the **Provider**.
Case Study
Translating into $\mathcal{CL}$ syntax

1. $\Box F_{P(s)}(fi)$

2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$

3. $\Box([d\&n](O(l) \land [l]\Diamond O(p\&p)))$

4. If the Client does not lower the Internet traffic immediately, then the Client will have to pay $3 \times [price]$.

5. The Client shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the Provider’s web page to the Client Relations Department of the Provider.
1. $\square F_{P(s)}(fi)$

2. $\square [h](\phi \Rightarrow O(p + (d\&n)))$

3. $\square ([d\&n](O(l) \land [l]\Diamond O(p\&p)))$

4. $\square ([d\&n \cdot \bar{l}]\Diamond O(p\&p\&p))$

5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**’s web page to the Client Relations Department of the **Provider**.
1. $\Box F_{P(s)}(fi)$

2. $\Box [h](\phi \Rightarrow O(p + (d\&n)))$

3. $\Box ([d\&n](O(l) \land [l]\diamond O(p\&p)))$

4. $\Box ([d\&n \cdot \bar{1}]\diamond O(p\&p\&p))$

5. $\Box ([o]O(sfD))$
$\phi = \text{the Internet traffic is high}$

$fi = \text{client supplies false information}$

$\text{to Client Relations Department}$

$h = \text{client increases Internet traffic}$

$\text{to high level}$

$p = \text{client pays [price]}$

$d = \text{client delays payment}$

$n = \text{client notifies by e-mail}$

$l = \text{client lowers the Int. traffic}$

$sfD = \text{client sends the Personal}$

$\text{Data Form to Client Relations}$

$\text{Department}$

$o = \text{provider activates the Internet}$

$\text{Service (it becomes operative)}$

$s = \text{provider suspends service}$
Case Study
Handcrafting the model

\[ \phi = \text{the Internet traffic is high} \]
\[ fi = \text{client supplies false information to Client Relations Department} \]
\[ h = \text{client increases Internet traffic to high level} \]
\[ p = \text{client pays [price]} \]
\[ d = \text{client delays payment} \]
\[ n = \text{client notifies by e-mail} \]
\[ l = \text{client lowers the Int. traffic} \]
\[ sfD = \text{client sends the Personal Data Form to Client Relations Department} \]
\[ o = \text{provider activates the Internet Service (it becomes operative)} \]
\[ s = \text{provider suspends service} \]
1. $\Box F_{P(s)}(fi)$
2. $\Box[h](\phi \implies O(p + (d \& n)))$
3. $\Box([d \& n](O(l) \land [l]O(p \& p)))$
4. $\Box([d \& n \cdot l]O(p \& p \& p))$
5. $\Box([o]O(sfD))$
“It is always the case that whenever the Internet traffic is high, if the client pays immediately, then the client is not obliged to pay again immediately afterward.”
Case Study
Verifying a property about client obligations

- “It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is not obliged to pay again immediately afterward”

- It fails!

![Diagram with states and transitions representing the property verification process.](image-url)
“It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is not obliged to pay again immediately afterward”

It fails!

We get a counter-example – Problem: state $s_4$
“It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is not obliged to pay again immediately afterward”

- It fails!
- We get a counter-example
  –Problem: state \( s_4 \)
- We modify the original contract to capture the above more precisely
“It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice”
“It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice”

It fails!
Case Study
Verifying a property about payment in case of increasing Internet traffic

- “It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice”
- It fails!
- Counter-example: From $s_4$ (φ holds), after $d \& n \cdot l$, it is possible to increase Internet traffic in state $s_7$, so neither $F(h)$ nor done$_{p\&p}$ hold
Case Study
Verifying a property about payment in case of increasing Internet traffic

“It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice”

It fails!

Counter-example: From $s_4$ ($\phi$ holds), after $d \& n \cdot l$, it is possible to increase Internet traffic in state $s_7$, so neither $F(h)$ nor $\text{done}_{p \& p}$ hold

Add to the original contract the clause above!
A formal specification language for contracts with semantics based on a variant of $\mu$-calculus
A formal specification language for contracts with semantics based on a variant of $\mu$-calculus

Use of model checking for reasoning about contracts:

1. Model checking to increase confidence in the correctness of the model wrt the original natural language contract
2. We identify problems in the original natural language contract or its interpretation in $CL$
3. We ensure certain desirable properties hold (and certain undesirable ones do not) for the signatories
Recent work

- Redesign $\mathcal{CL}$ (changes going fast!) [C. Prisacariu]
- Branching semantics [C. Prisacariu]
- Trace semantics (run-time monitoring) [C. Prisacariu & M. Kyas]
- Case study (CoCoME) [S. Fenech & G. Pace]

On-going work

- Automate the model checking process […]
- Encoding into LTL to check inconsistencies [S. Fenech & G. Pace]
Current and Further Work

Further work

- “Normative” automata
- Develop a proof system
- Internal vs external operations
  - Obligation on sequences vs sequence of obligations
  - Kleene star vs Until
  - $+ \text{ vs } \oplus$
- Add time
  - Timed $\mu$-calculus, TCTL, ...? Associated with actions or formulae?
  - Durations, time stamps, beginning and end, dates, ...?
- Negotiation
  - Maude
- Applications (besides SOA)
  - Component-based development
  - Fault-tolerant systems
  - Compensable transactions
Links and Papers

- **COSoDIS**: “Contract-Oriented Software Development for Internet Services” – A Nordunet3 project
  (http://www.ifi.uio.no/cosodis/)

- **FLACOS’07** – 1st Workshop on Formal Languages and Analysis of Contract-Oriented Software (http://www.ifi.uio.no/flacos07/)
  - Oslo, 9-10 October 2007


Thank you!