Model Checking Contracts
A case study

Gordon Pace
gordon.pace@um.edu.mt

Cristian Prisacariu
cristi@ifi.uio.no

Gerardo Schneider
gerardo@ifi.uio.no

Department of Informatics,
University of Oslo

ATVA'07
Tokyo, Japan
October 22-25, 2007

Contracts

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

Contracts

- "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

- 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

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

Outline

1. The Contract Language $\mathcal{CL}$
2. Model Checking Contracts
3. Final Remarks

Outline

1. The Contract Language $\mathcal{CL}$
2. Model Checking Contracts
3. Final Remarks

The Contract Specification Language $\mathcal{CL}$

$$
\begin{array}{rcl}
\mathcal{C}ontract & := & \mathcal{D}\ ;\ \mathcal{C} \\
\mathcal{C} & := & \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \wedge \mathcal{C} \mid [\alpha]\mathcal{C} \mid \langle\alpha\rangle\mathcal{C} \mid \mathcal{C}\,\mathcal{U}\,\mathcal{C} \mid \bigcirc\mathcal{C} \mid \Box\mathcal{C} \\
\mathcal{C}_O & := & O(\alpha) \mid \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P & := & P(\alpha) \mid \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F & := & F(\alpha) \mid \mathcal{C}_F \vee [\alpha]\mathcal{C}_F
\end{array}
$$

- $O(\alpha)$, $P(\alpha)$, $F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- α are actions given in the definition part $\mathcal{D}$
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrency
  - $\phi?$ test
- $\wedge$, $\vee$, 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

The Contract Specification Language $\mathcal{CL}$

$$
\begin{array}{rcl}
\mathcal{C}ontract & := & \mathcal{D}\ ;\ \mathcal{C} \\
\mathcal{C} & := & \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \wedge \mathcal{C} \mid [\alpha]\mathcal{C} \mid \langle\alpha\rangle\mathcal{C} \mid \mathcal{C}\,\mathcal{U}\,\mathcal{C} \mid \bigcirc\mathcal{C} \mid \Box\mathcal{C} \\
\mathcal{C}_O & := & O(\alpha) \mid \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P & := & P(\alpha) \mid \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F & := & F(\alpha) \mid \mathcal{C}_F \vee [\alpha]\mathcal{C}_F
\end{array}
$$

- $O(\alpha)$, $P(\alpha)$, $F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- α are actions given in the definition part $\mathcal{D}$
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrency
  - $\phi?$ test
- $\wedge$, $\vee$, 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

The Contract Specification Language $\mathcal{CL}$

$$
\begin{array}{rcl}
\mathcal{C}ontract & := & \mathcal{D}\ ;\ \mathcal{C} \\
\mathcal{C} & := & \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \wedge \mathcal{C} \mid [\alpha]\mathcal{C} \mid \langle\alpha\rangle\mathcal{C} \mid \mathcal{C}\,\mathcal{U}\,\mathcal{C} \mid \bigcirc\mathcal{C} \mid \Box\mathcal{C} \\
\mathcal{C}_O & := & O(\alpha) \mid \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P & := & P(\alpha) \mid \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F & := & F(\alpha) \mid \mathcal{C}_F \vee [\alpha]\mathcal{C}_F
\end{array}
$$

- $O(\alpha)$, $P(\alpha)$, $F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- α are actions given in the definition part $\mathcal{D}$
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrency
  - $\phi?$ test
- $\wedge$, $\vee$, 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

The Contract Specification Language $\mathcal{CL}$

$$
\begin{array}{rcl}
\mathcal{C}ontract & := & \mathcal{D}\ ;\ \mathcal{C} \\
\mathcal{C} & := & \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \wedge \mathcal{C} \mid [\alpha]\mathcal{C} \mid \langle\alpha\rangle\mathcal{C} \mid \mathcal{C}\,\mathcal{U}\,\mathcal{C} \mid \bigcirc\mathcal{C} \mid \Box\mathcal{C} \\
\mathcal{C}_O & := & O(\alpha) \mid \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P & := & P(\alpha) \mid \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F & := & F(\alpha) \mid \mathcal{C}_F \vee [\alpha]\mathcal{C}_F
\end{array}
$$

- $O(\alpha)$, $P(\alpha)$, $F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- α are actions given in the definition part $\mathcal{D}$
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrency
  - $\phi?$ test
- $\wedge$, $\vee$, 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

The Contract Specification Language $\mathcal{CL}$

$$
\begin{array}{rcl}
\mathcal{C}ontract & := & \mathcal{D}\ ;\ \mathcal{C} \\
\mathcal{C} & := & \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \wedge \mathcal{C} \mid [\alpha]\mathcal{C} \mid \langle\alpha\rangle\mathcal{C} \mid \mathcal{C}\,\mathcal{U}\,\mathcal{C} \mid \bigcirc\mathcal{C} \mid \Box\mathcal{C} \\
\mathcal{C}_O & := & O(\alpha) \mid \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P & := & P(\alpha) \mid \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F & := & F(\alpha) \mid \mathcal{C}_F \vee [\alpha]\mathcal{C}_F
\end{array}
$$

- $O(\alpha)$, $P(\alpha)$, $F(\alpha)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- α are actions given in the definition part $\mathcal{D}$
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrency
  - $\phi?$ test
- $\wedge$, $\vee$, 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

More on the Contract Language

CTD and CTP

- Expressing contrary-to-duty (CTD)

$$O_{\mathcal{C}}(\alpha) = O(\alpha) \wedge [\overline{\alpha}]\mathcal{C}$$

More on the Contract Language

CTD and CTP

- Expressing contrary-to-duty (CTD)

$$O_{\mathcal{C}}(\alpha) = O(\alpha) \wedge [\overline{\alpha}]\mathcal{C}$$

- Expressing contrary-to-prohibition (CTP)

$$F_{\mathcal{C}}(\alpha) = F(\alpha) \wedge [\alpha]\mathcal{C}$$

$\mathcal{CL}$ Semantics

$\mathcal{C}\mu$ – A variant of the modal μ-calculus

- Translation into a variant of μ-calculus ($\mathcal{C}\mu$)
- The syntax of the $\mathcal{C}\mu$ logic

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

Main differences with respect to the classical μ-calculus:

1. P_c is set of propositional constants O_a and $\mathcal{F}_a$, one for each basic action a
2. Multisets of basic actions: i.e. $\gamma = \{a, a, b\}$ is a label

$\mathcal{CL}$ Semantics

$\mathcal{C}\mu$ – A variant of the modal μ-calculus

- Translation into a variant of μ-calculus ($\mathcal{C}\mu$)
- The syntax of the $\mathcal{C}\mu$ logic
  $\varphi := P \mid Z \mid P_c \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid [\gamma]\varphi \mid \mu Z.\varphi(Z)$

Main differences with respect to the classical μ-calculus:

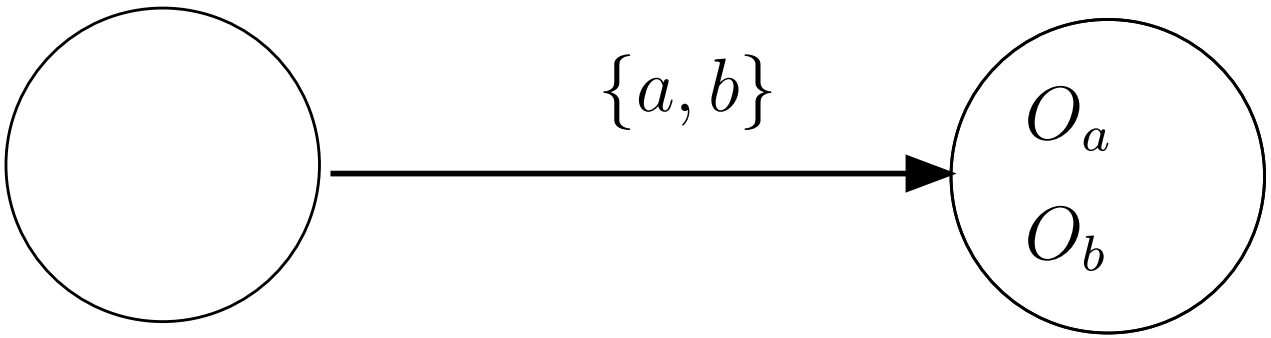
1. P_c is set of propositional constants O_a and $\mathcal{F}_a$, one for each basic action a
2. Multisets of basic actions: i.e. $\gamma = \{a, a, b\}$ is a label

- Obligation

$$f^{\mathcal{T}}(O(a\&b)) = \langle\{a, b\}\rangle(O_a \wedge O_b)$$

- Obligation

$$f^{\mathcal{T}}(O(a\&b)) = \langle\{a, b\}\rangle(O_a \wedge O_b)$$



$O(a\&b)$

Outline

1 The Contract Language $\mathcal{CL}$

2 Model Checking Contracts

3 Final Remarks

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

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

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

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

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

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

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

Case Study

A Contract Example

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 $(2 * [price])$.
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**.
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;

Case Study

A Contract Example

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 $(2 * [price])$.
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**.
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;

Case Study

Translating into $\mathcal{CL}$ syntax

1. $\Box F(fi)$

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 $(2 * [price])$.
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**.
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;

Case Study

Translating into $\mathcal{CL}$ syntax

1. $\Box F(fi)$

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 $(2 * [price])$.
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**.
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;

Case Study

Translating into $\mathcal{CL}$ syntax

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

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 $(2 * [price])$.
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**.

Case Study

Translating into $\mathcal{CL}$ syntax

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 $(2 * [price])$.
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**.

Case Study

Translating into $\mathcal{CL}$ syntax

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

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

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

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**.

Case Study

Translating into $\mathcal{CL}$ syntax

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

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

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

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

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**.

Case Study

Translating into $\mathcal{CL}$ syntax

1. $\Box F_{P(s)}(fi)$
2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

Case Study

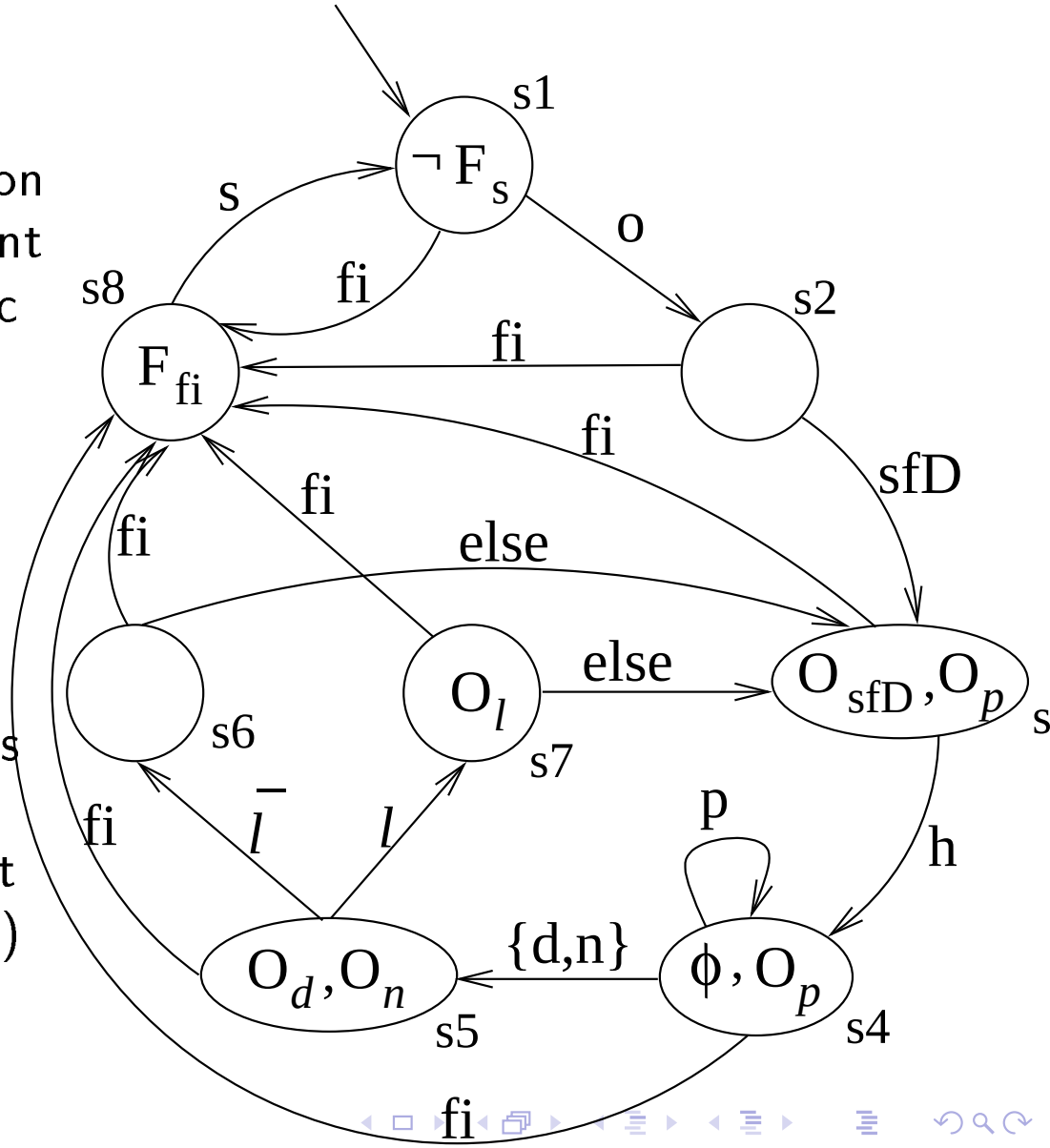
Handcrafting the model

ϕ = the Internet traffic is high
fi = client supplies false information to Client Relations Department
h = client increases Internet traffic to *high* level
p = client pays [price]
d = client delays payment
n = client notifies by e-mail
l = client lowers the Int. traffic
sfD = client sends the Personal Data Form to Client Relations Department
o = provider activates the Internet Service (it becomes operative)
s = provider suspends service

Case Study

Handcrafting the model

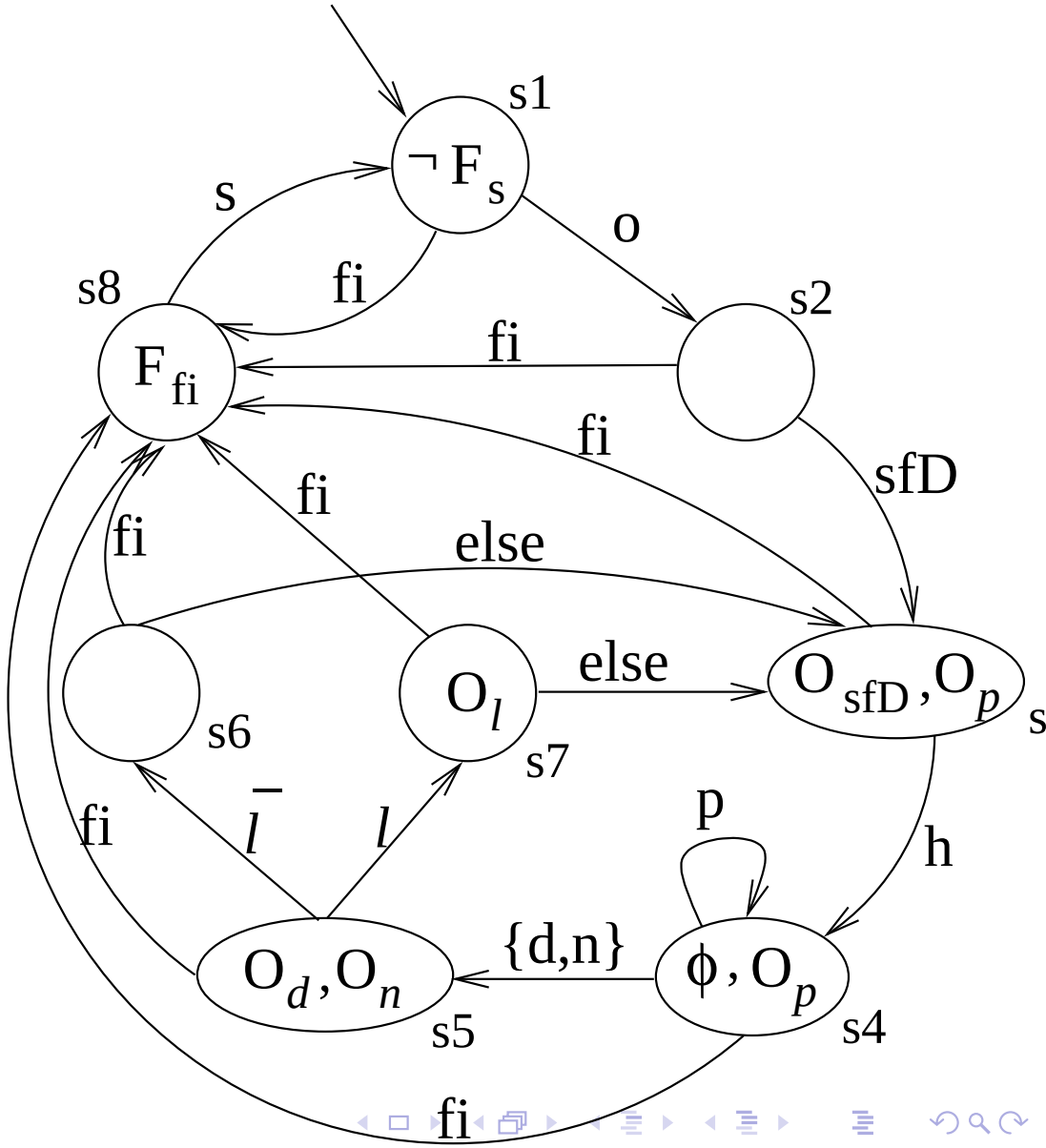
ϕ = the Internet traffic is high
fi = client supplies false information to Client Relations Department
h = client increases Internet traffic to *high* level
p = client pays [price]
d = client delays payment
n = client notifies by e-mail
l = client lowers the Int. traffic
sfD = client sends the Personal Data Form to Client Relations Department
o = provider activates the Internet Service (it becomes operative)
s = provider suspends service



Case Study

Checking the contract on the model

1. $\Box F_{P(s)}(fi)$
2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

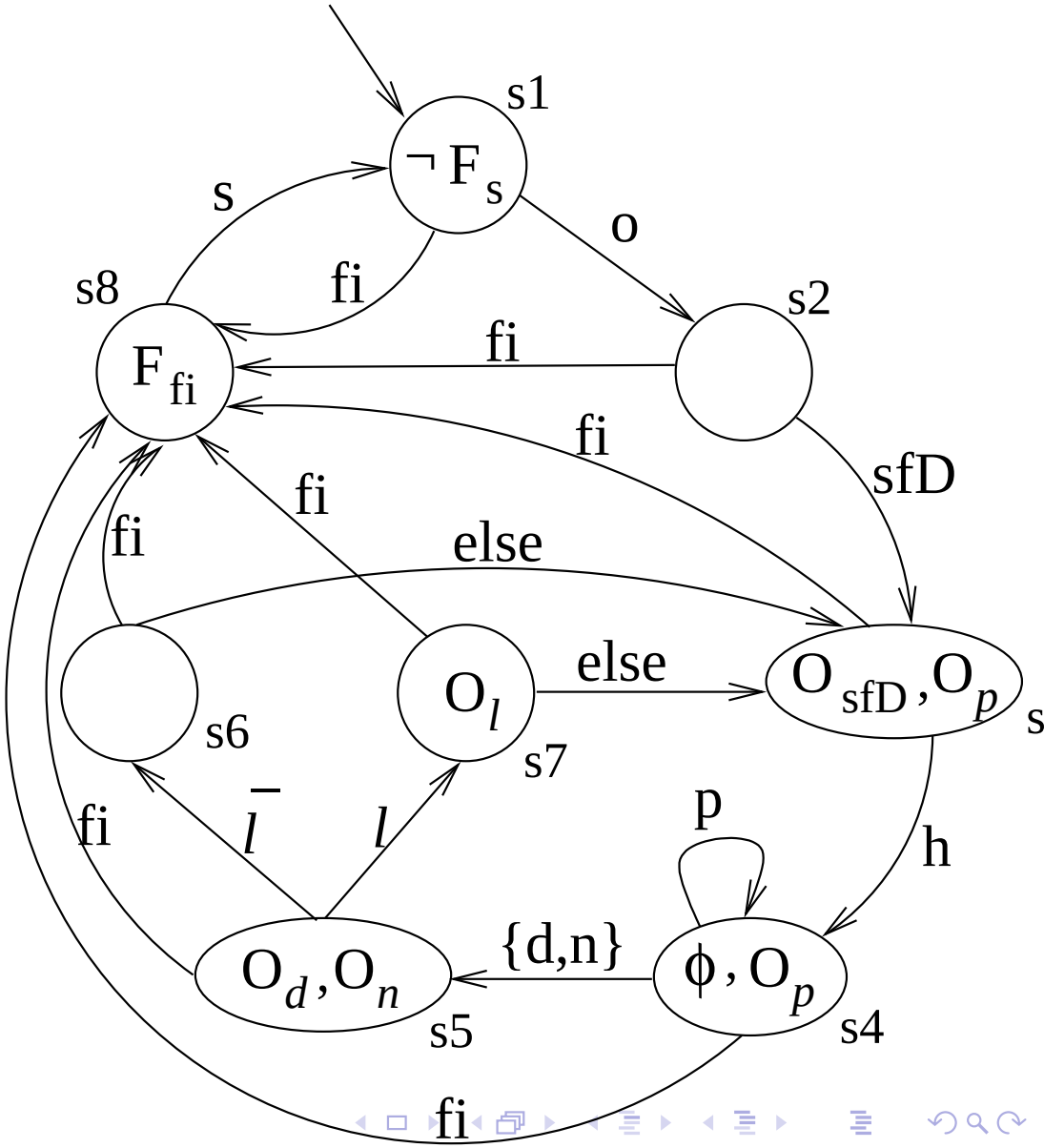


Case Study

Checking the contract on the model

1. $\Box F_{P(s)}(fi)$
2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

1, 2, and 4: OK

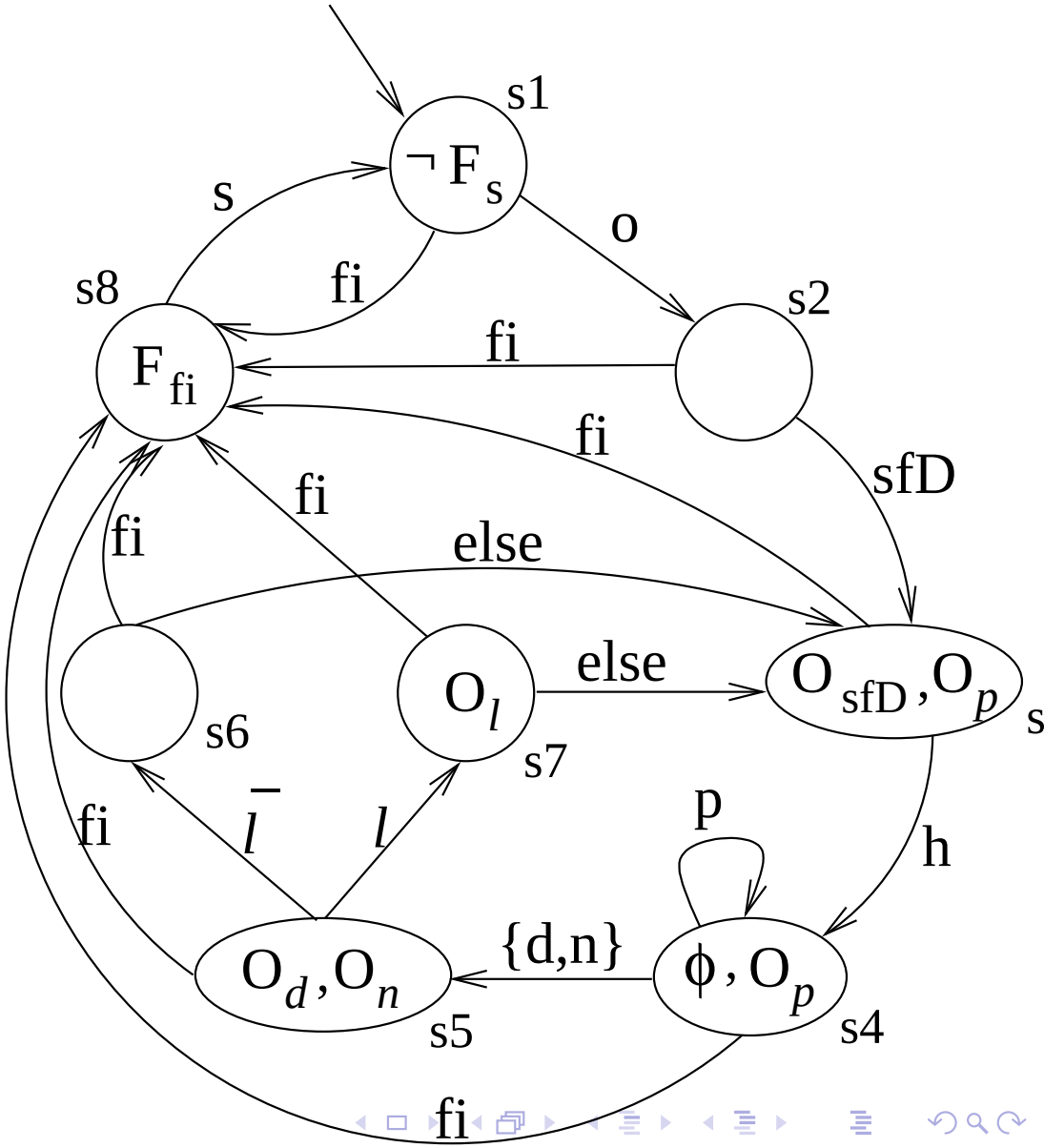


Case Study

Checking the contract on the model

1. $\Box F_{P(s)}(fi)$
2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

1, 2, and 4: OK
3 and 5: FAIL!



Case Study

Checking the contract on the model (cont.)

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Case Study

Checking the contract on the model (cont.)

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

Case Study

Checking the contract on the model (cont.)

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

Failure of 5. $(\Box([o]O(sfD)))$

- The system should become operative only once

Case Study

Checking the contract on the model (cont.)

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

Failure of 5. $(\Box([o]O(sfD)))$

- The system should become operative only once

1. We rewrite the original contract
2. This is formulated in $\mathcal{CL}$, written in NuSMV, and it model checks!

Case Study

Checking the contract on the model (cont.)

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

Failure of 5. $(\Box([o]O(sfD)))$

- The system should become operative only once

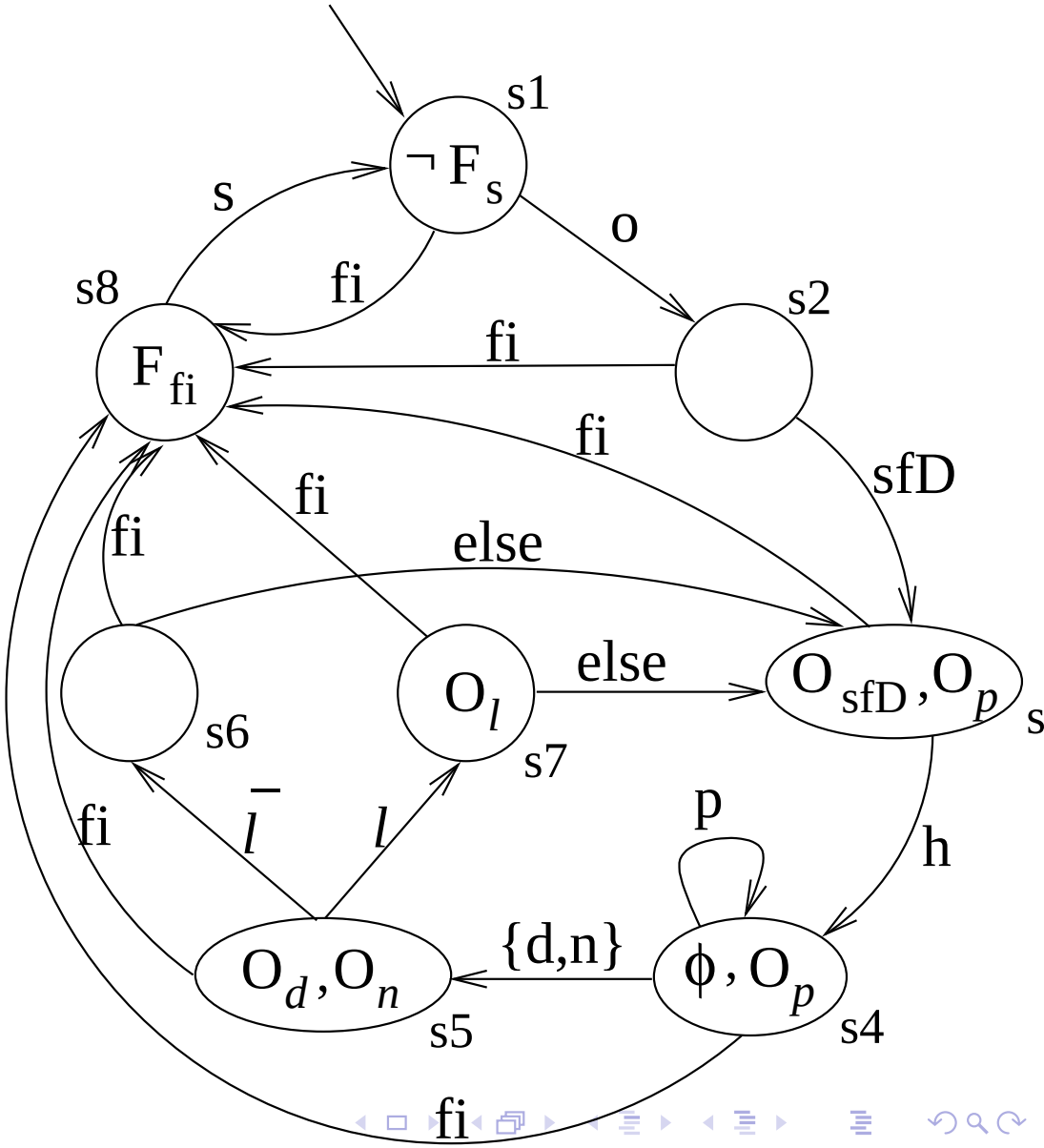
1. We rewrite the original contract
2. This is formulated in $\mathcal{CL}$, written in NuSMV, and it model checks!

'Failure' on the original contract!

Case Study

Verifying a property about client obligations

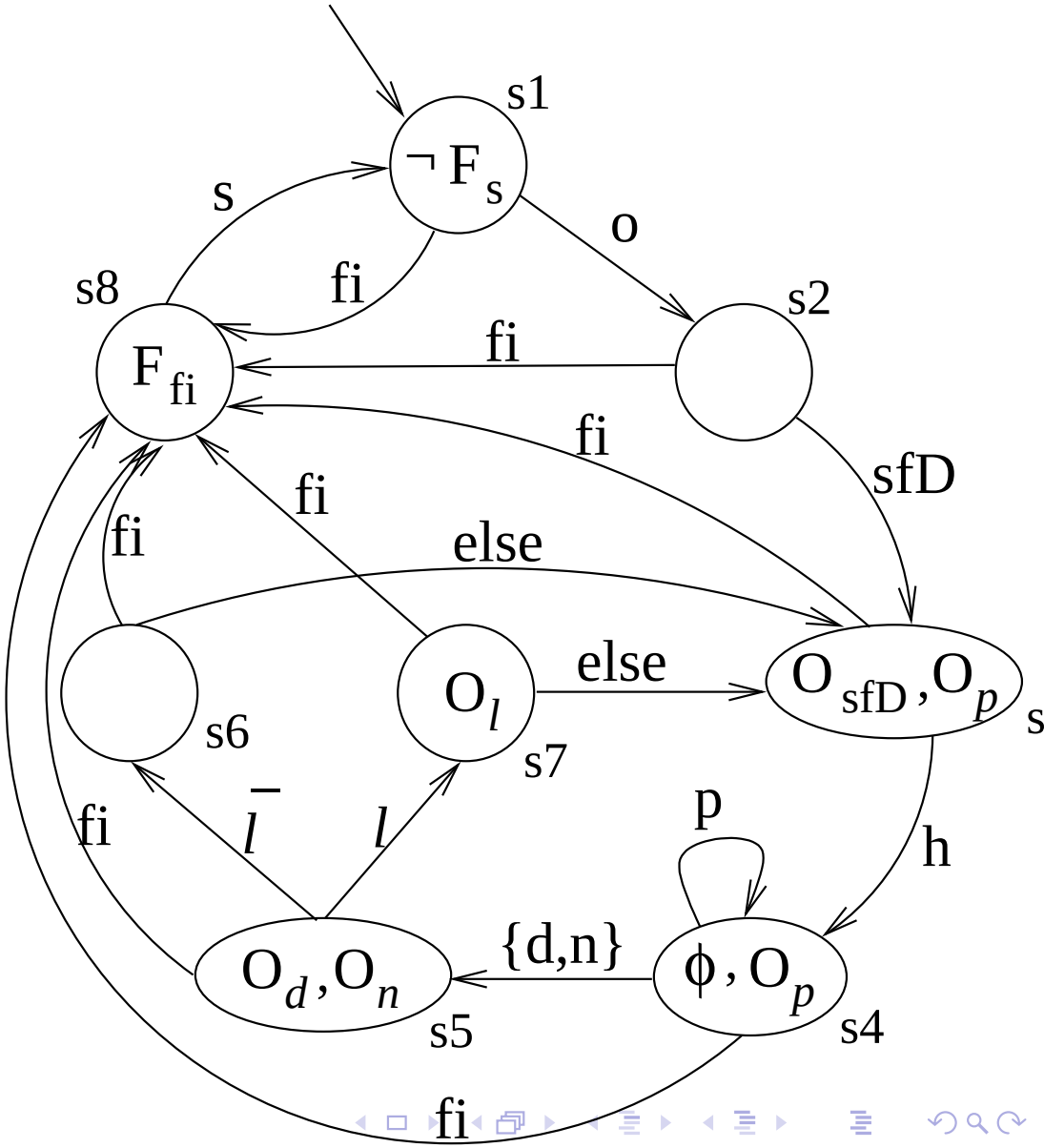
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"



Case Study

Verifying a property about client obligations

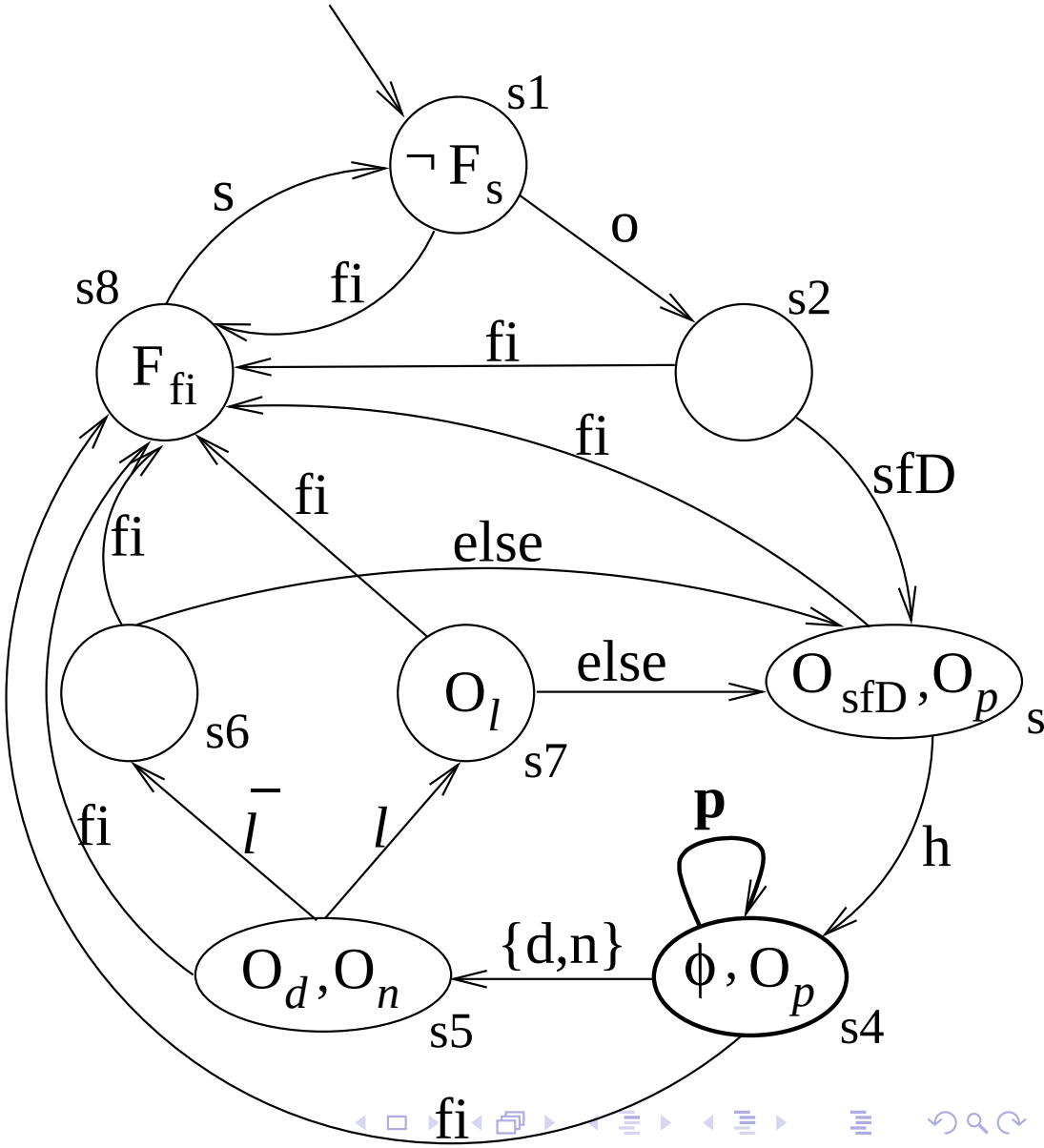
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"
- It fails!



Case Study

Verifying a property about client obligations

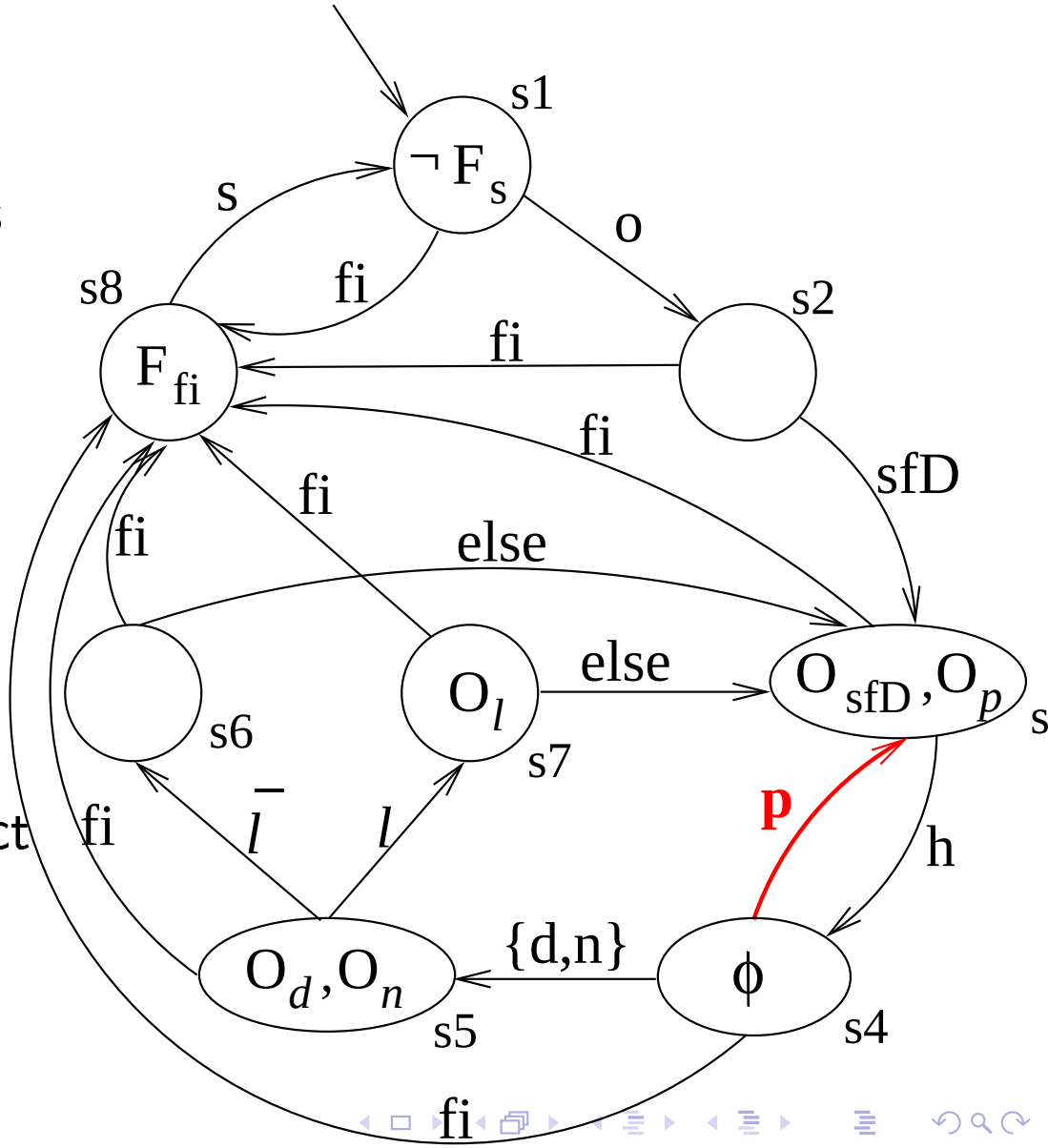
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"
- It fails!
- We get a counter-example –Problem: state *s*4



Case Study

Verifying a property about client obligations

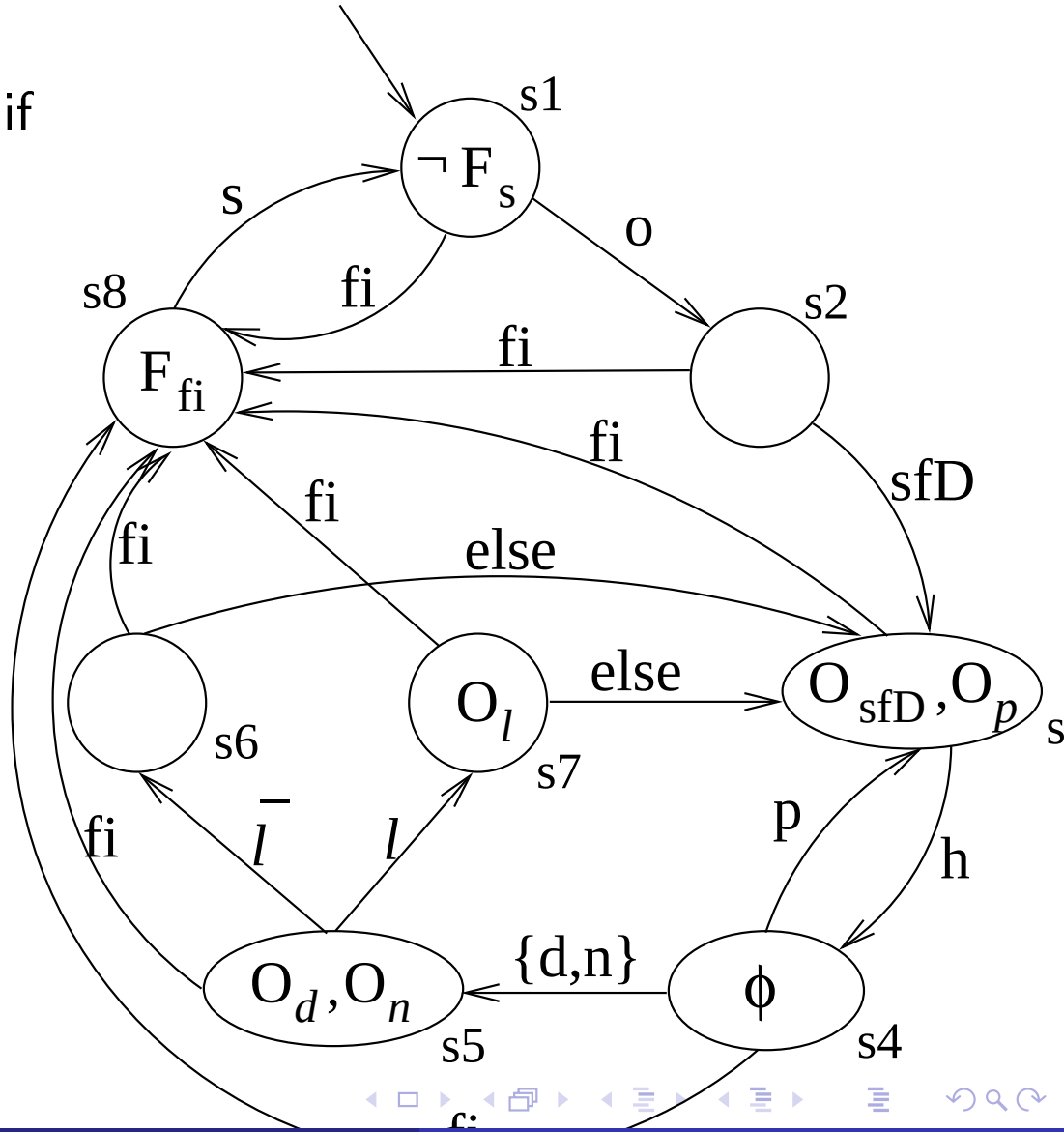
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"
- It fails!
- We get a counter-example –Problem: state *s*4
- We modify the original contract to capture the above more precisely



Case Study

Verifying a property about payment in case of increasing Internet traffic

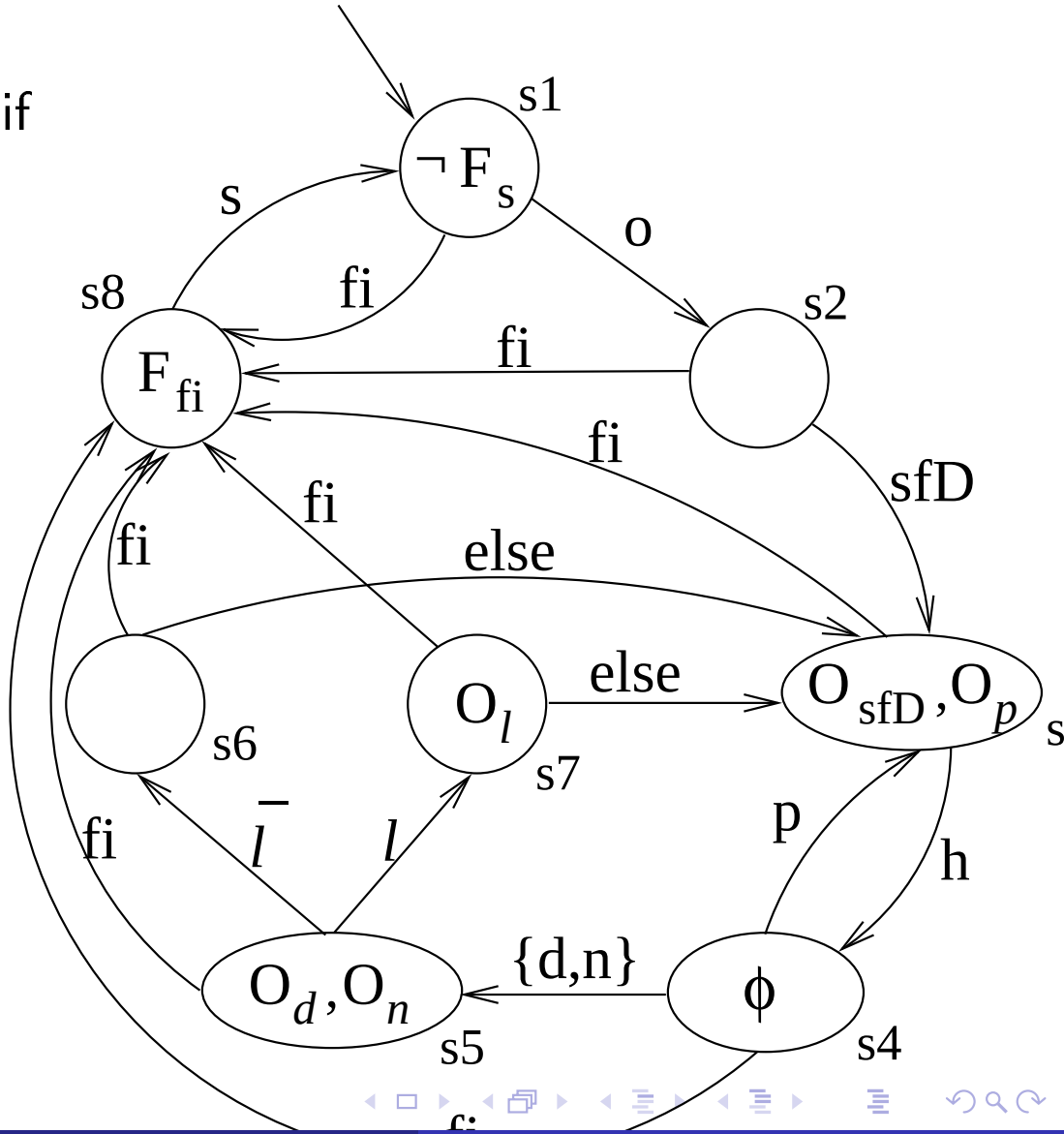
- "It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice"



Case Study

Verifying a property about payment in case of increasing Internet traffic

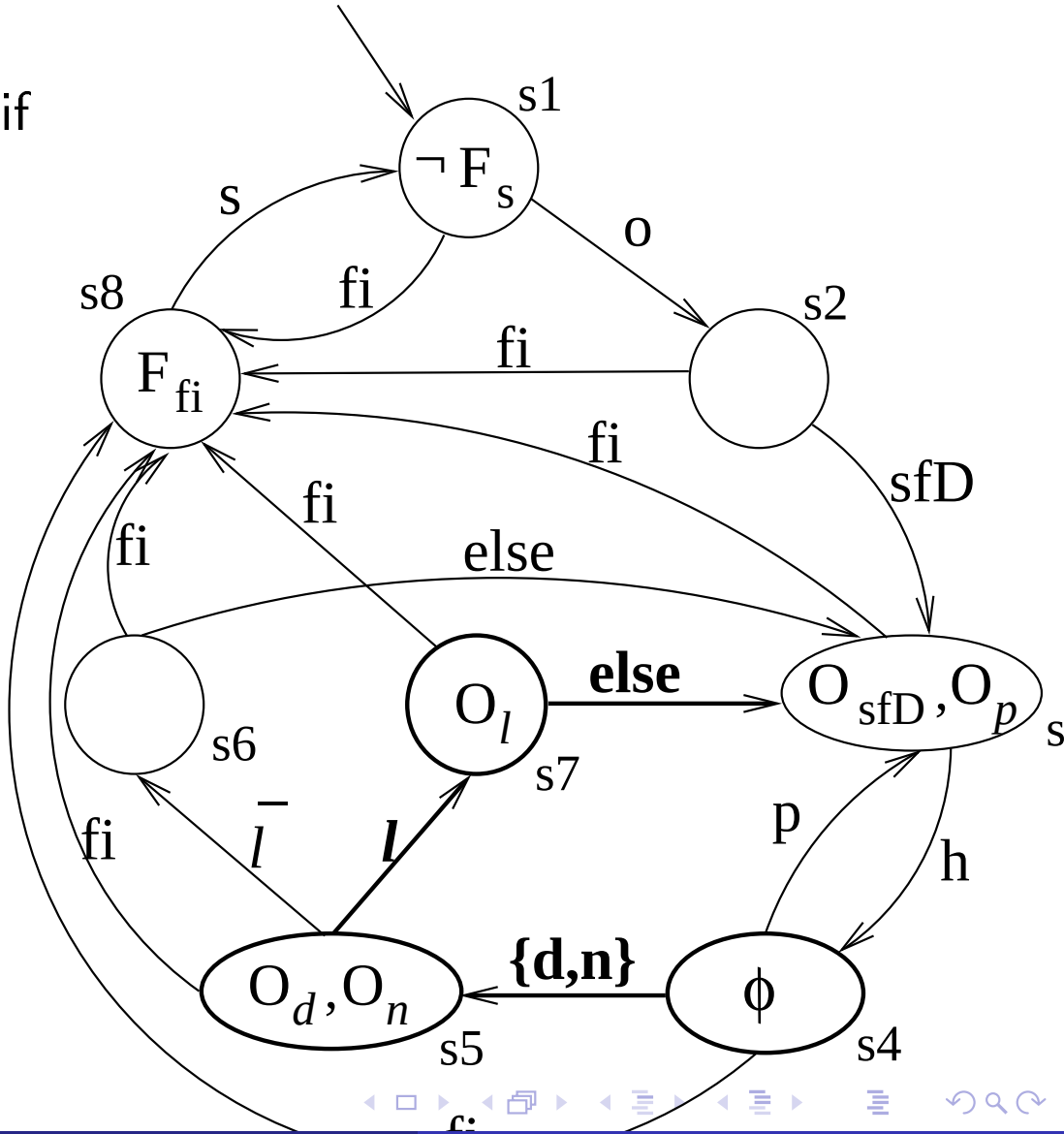
- "It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice"
- It fails!



Case Study

Verifying a property about payment in case of increasing Internet traffic

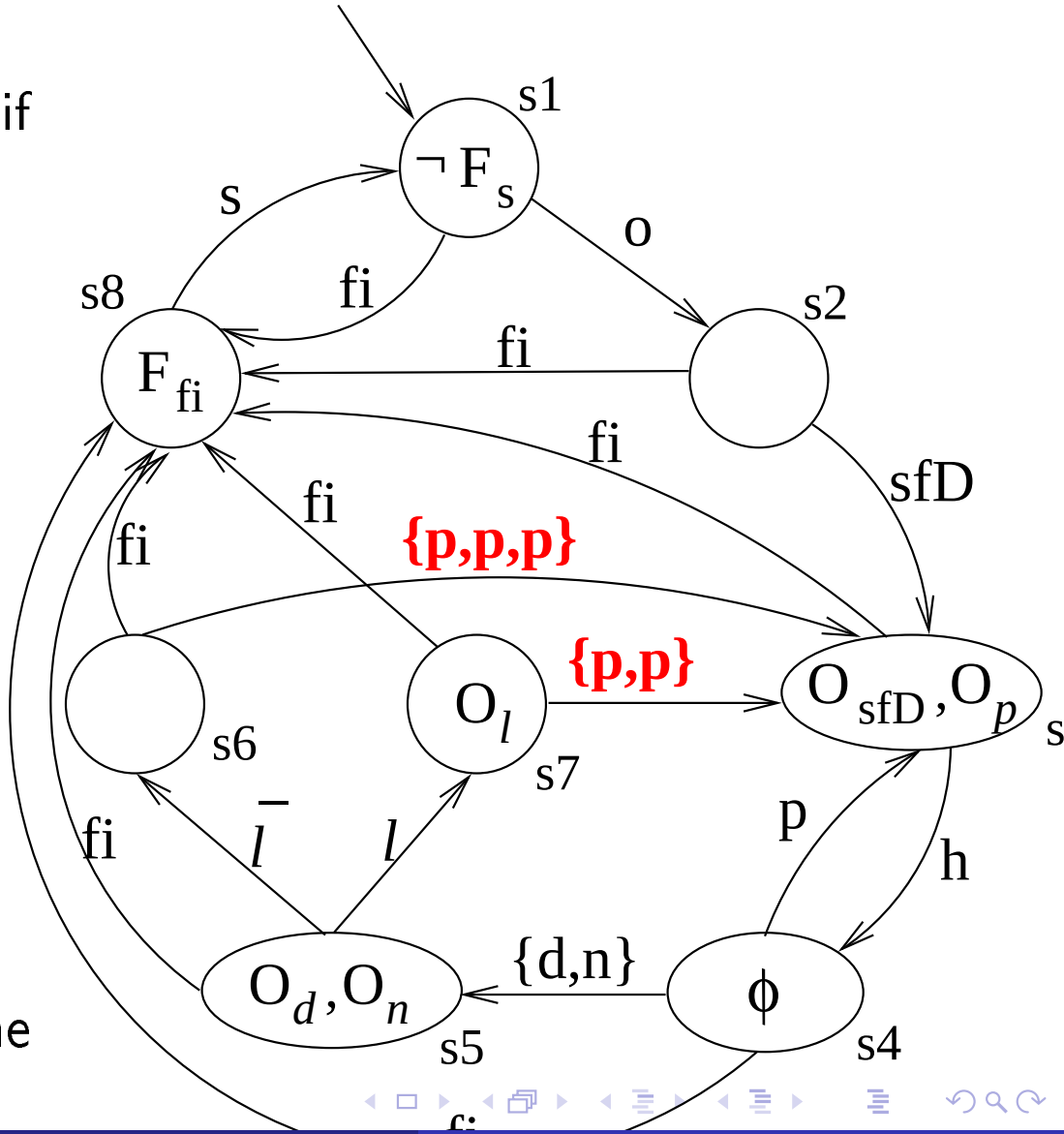
- "It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice"
- It fails!
- Counter-example: From s_4 (ϕ holds), after $d\&n \cdot l$,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



Outline

1. The Contract Language $\mathcal{CL}$
2. Model Checking Contracts
3. Final Remarks

Model Checking Contracts

- Initial ideas on how to model check contracts

Based on:

- A formal specification language for contracts with semantics based on a variant of μ-calculus

Model Checking Contracts

- Initial ideas on how to model check contracts

Based on:

- A formal specification language for contracts with semantics based on a variant of μ-calculus

Use of model checking for reasoning about contracts:

1. We use model checking to increase our confidence in the correctness of the model with respect to the original natural language contract
2. By finding errors in the model, we identify problems in the original natural language contract or its interpretation in $\mathcal{CL}$
3. We enable the signatories to safeguard their interests by ensuring certain desirable properties hold (and certain undesirable ones do not)

Current and Further Work

Currently:

- Direct semantics: "Normative" automata
- Redesign $\mathcal{CL}$
- Automate the model checking process

Current and Further Work

Currently:

- Direct semantics: "Normative" automata
- Redesign $\mathcal{CL}$
- Automate the model checking process

Further work:

- Develop a proof system
- Internal vs external operations
- Add time
- Case studies
- Explore how to extract a contract monitor (?!)

Thank you!

Links and Papers

- C. Prisacariu and G. Schneider. A formal language for electronic contracts. In FMOODS'07, vol. 4468 of LNCS, pages 174-189, June 2007

- COSoDIS: "Contract-Oriented Software Development for Internet Services" –A Nordunet3 project (`http://folk.uio.no/gerardo/nordunet3/index.shtml`)

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