On the Specification of Full Contracts

Stephen Fenech[1] Joseph Okika[2] Gordon Pace[1] Anders Ravn[2]
Gerardo Schneider[3]

sfen002@um.edu.mt ojc@cs.aau.dk gordon.pace@um.edu.mt apr@cs.aau.dk
gerardo@ifi.uio.no

[1]Dept. of Computer Science – University of Malta, Malta

[2]Dept. of Computer Science – Aalborg University, Denmark

[3]Dept. of Informatics – University of Oslo, Norway

FESCA, 28 March 2009 – York, UK

Full Contracts

- A contract is a binding agreement between two or more entities (enforceable by law)
- Use contracts to regulate interactions in concurrent/distributed systems
  - Components, services, etc
- Different notions (or *levels*) of contracts
  - Static interfaces
  - Behavioural interfaces
  - *Design-by-contract* (pre-, post-conditions, invariants, etc)
  - Quality-of-Service
  - 'Social' contracts
  - Deontic e-contracts

Full Contracts

- A contract is a binding agreement between two or more entities (enforceable by law)
- Use contracts to regulate interactions in concurrent/distributed systems
  - Components, services, etc
- Different notions (or *levels*) of contracts
  - Static interfaces
  - Behavioural interfaces
  - *Design-by-contract* (pre-, post-conditions, invariants, etc)
  - Quality-of-Service
  - 'Social' contracts
  - Deontic e-contracts

Full Contracts

- A contract is a binding agreement between two or more entities (enforceable by law)
- Use contracts to regulate interactions in concurrent/distributed systems
  - Components, services, etc
- Different notions (or *levels*) of contracts
  - Static interfaces
  - Behavioural interfaces
  - *Design-by-contract* (pre-, post-conditions, invariants, etc)
  - Quality-of-Service
  - 'Social' contracts
  - Deontic e-contracts

Full Contracts

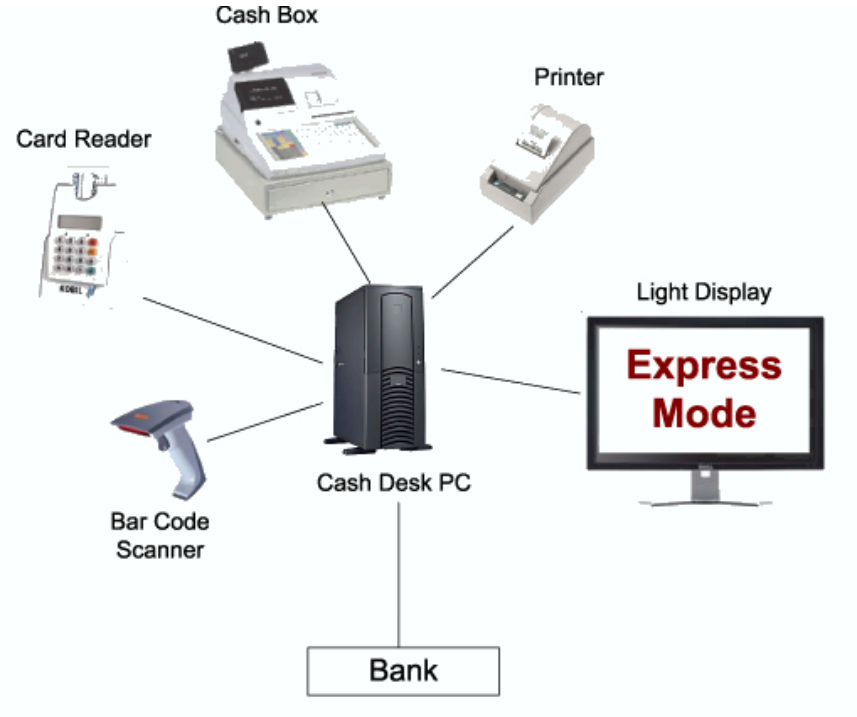
- A contract is a binding agreement between two or more entities (enforceable by law)
- Use contracts to regulate interactions in concurrent/distributed systems
  - Components, services, etc
- Different notions (or *levels*) of contracts
  - Static interfaces
  - Behavioural interfaces
  - *Design-by-contract* (pre-, post-conditions, invariants, etc)
  - Quality-of-Service
  - 'Social' contracts
  - Deontic e-contracts
- Full contracts: Normal and exceptional behaviour
  - Exceptions, compensations, tolerance to faults, penalties, etc

Objectives of Our Work

1. Specification of CoCoME (use cases 1-8) using $\mathcal{CL}$
   - $\mathcal{CL}$ is contract language based on deontic logic
   - It allows the specification of obligations, permissions and prohibitions, and the penalties in case of violations
2. To compare suitability of operational and logical approaches to specify full contracts on a well-known case study (CoCoME)
   - Operational
     - rCOS (Relational Calculus of Object and Component Systems –CSP implementation)
   - Logical
     - Deontic-logic based language ($\mathcal{CL}$)
     - Temporal logics

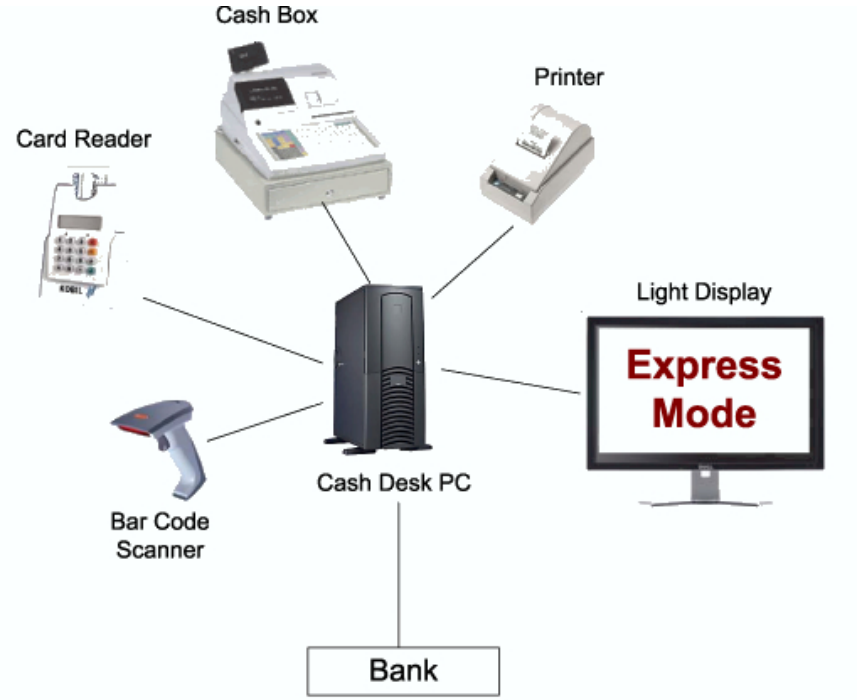
CoCoME: Common Component Modelling Example

- Trading System to handle sales and inventory of a store chain
- 8 use cases



CoCoME: Common Component Modelling Example

- Trading System to handle sales and inventory of a store chain
- 8 use cases
- Use case 1
  - How a sale is processed
- Use case 2
  - How a cash desk switches to express mode, restricting total number of customer items
- We focus on the behavioural aspects of the use cases
  - Prop1, Prop2, Prop3



Definition (CSP)

- CSP (rCOS)

$$P \quad ::= \quad Stop \mid a \rightarrow P \mid P[]P \mid P \sqcap P \mid X$$

Definition (CSP)

- CSP (rCOS)

$$P \quad ::= \quad Stop \mid a \rightarrow P \mid P[]P \mid P \sqcap P \mid X$$

Definition (Temporal Logics)

- LTL

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{G}\varphi \mid \mathbf{F}\varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi$$

- CTL

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{AG}\varphi \mid \mathbf{AF}\varphi \mid \mathbf{AX}\varphi \mid \varphi\mathbf{AU}\varphi \mid \mathbf{EG}\varphi \mid \mathbf{EF}\varphi \mid \mathbf{EX}\varphi \mid \varphi\mathbf{EU}\varphi$$

Definition ($\mathcal{CL}$ Syntax)

$$
\begin{array}{rcl}
C & := & C_O \mid C_P \mid C_F \mid C \wedge C \mid [\beta]C \mid \top \mid \bot \\
C_O & := & \mathbb{O}_C(\alpha) \mid C_O \oplus C_O \\
C_P & := & \mathbb{P}(\alpha) \mid C_P \oplus C_P \\
C_F & := & \mathbb{F}_C(\alpha) \mid C_F \vee [\alpha]C_F \\
\alpha & := & 0 \mid 1 \mid a \mid \alpha \& \alpha \mid \alpha ; \alpha \mid \alpha + \alpha \qquad \beta := 0 \mid 1 \mid a \mid \beta \& \beta \mid \beta ; \beta \mid \beta + \beta \mid \beta^*
\end{array}
$$

Definition ($\mathcal{CL}$ Syntax)

$$
\begin{array}{rcl}
C & := & C_O \mid C_P \mid C_F \mid C \wedge C \mid [\beta]C \mid \top \mid \bot \\
C_O & := & \mathbb{O}_C(\alpha) \mid C_O \oplus C_O \\
C_P & := & \mathbb{P}(\alpha) \mid C_P \oplus C_P \\
C_F & := & \mathbb{F}_C(\alpha) \mid C_F \vee [\alpha]C_F \\
\alpha & := & 0 \mid 1 \mid a \mid \alpha \& \alpha \mid \alpha ; \alpha \mid \alpha + \alpha \quad \beta := 0 \mid 1 \mid a \mid \beta \& \beta \mid \beta ; \beta \mid \beta + \beta \mid \beta^*
\end{array}
$$

Example (Specification of Prop1)

If pay with card, three allowed attempts to enter pin code; otherwise pay with cash, or return goods

$$\Box[\text{cardPay}]\ \mathbb{O}_{\psi_1}(\mathit{correctPin})$$

where $\psi_1 = \mathbb{O}_{\psi_2}(\mathit{correctPin})$, with $\psi_2 = \mathbb{O}_{\mathbb{O}(\mathit{cashPay}+\mathit{returnItems})}(\mathit{correctPin})$

Specification of CoCoME Use Cases 1 and 2

Specific clauses to be specified

Prop1: Pay by cash: obligation to swipe the card + correct pin

- Incorrect pin: two more allowed attempts
- After 3 incorrect pins: obligation to pay cash
- No cash: give up the goods

Specification of CoCoME Use Cases 1 and 2

Specific clauses to be specified

Prop1: Pay by cash: obligation to swipe the card + correct pin

- Incorrect pin: two more allowed attempts
- After 3 incorrect pins: obligation to pay cash
- No cash: give up the goods

Prop2: Normal mode: the cashier may switch to express mode

- If in the last hour 50% of the sales had less than eight items (*)
- In express mode: cashier obliged to eventually go to normal mode
- If (*) holds infinitely often, then the cashier should change to express mode infinitely often

Specification of CoCoME Use Cases 1 and 2

Specific clauses to be specified

Prop1: Pay by cash: obligation to swipe the card + correct pin

- Incorrect pin: two more allowed attempts
- After 3 incorrect pins: obligation to pay cash
- No cash: give up the goods

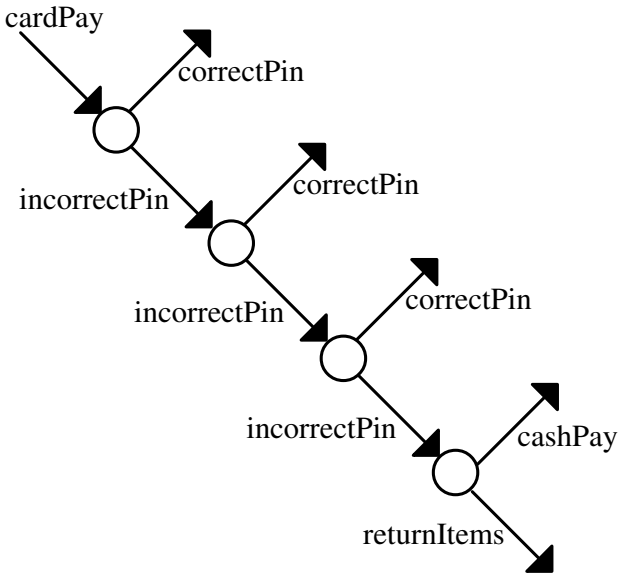
Prop2: Normal mode: the cashier may switch to express mode

- If in the last hour 50% of the sales had less than eight items (*)
- In express mode: cashier obliged to eventually go to normal mode
- If (*) holds infinitely often, then the cashier should change to express mode infinitely often

Prop3: In express mode: cashier obliged to service customers with less than eight items

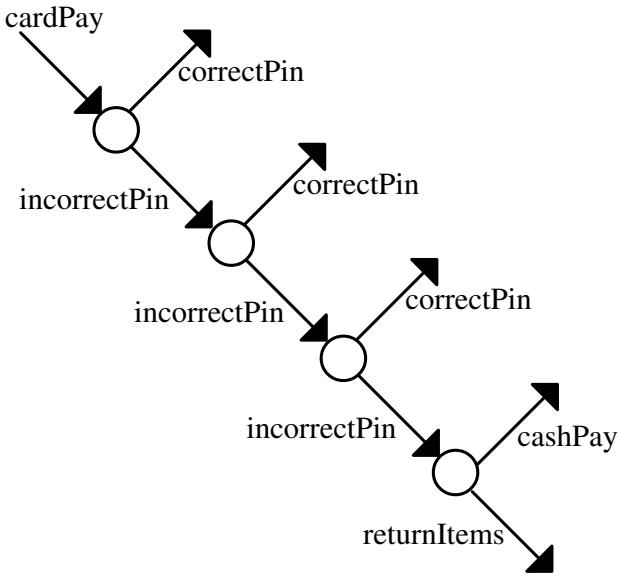
- If customer with more than eight items: cashier decides whether to service the client

Specification of Prop1



- Payment with credit card
- Three allowed attemps to enter correct pin

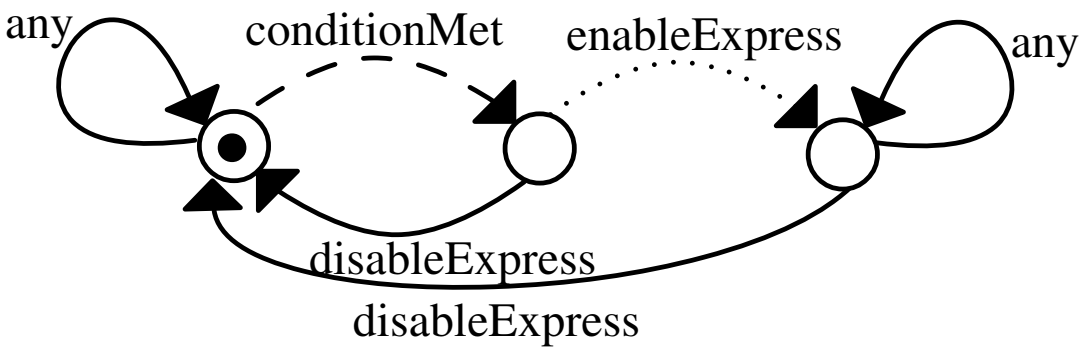
Specification of Prop1



- Payment with credit card
- Three allowed attemps to enter correct pin

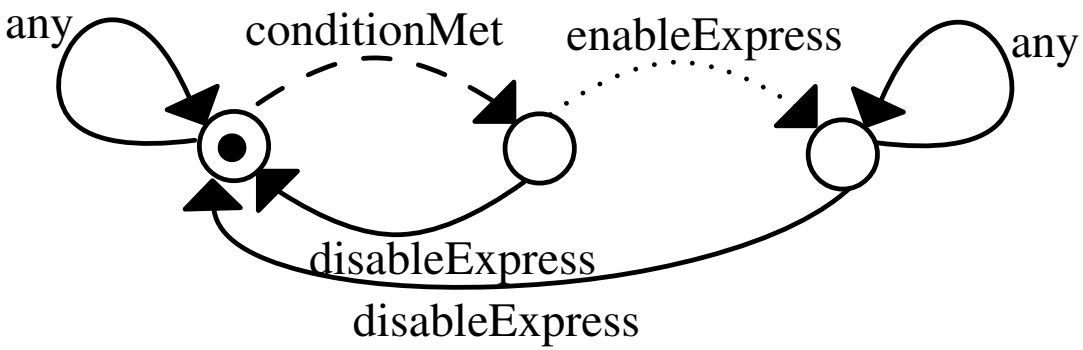
- CSP: Specification of normal case + refinement to capture exceptional behaviour
  - Possible but intricate branching
- Can be described in both CTL and LTL
- Can be described in $\mathcal{CL}$ (using CTDs)

Specification of Prop2



- Permission to switch from normal to express mode
- Obligation to come back to normal mode
- Fairness constraint between normal and express mode

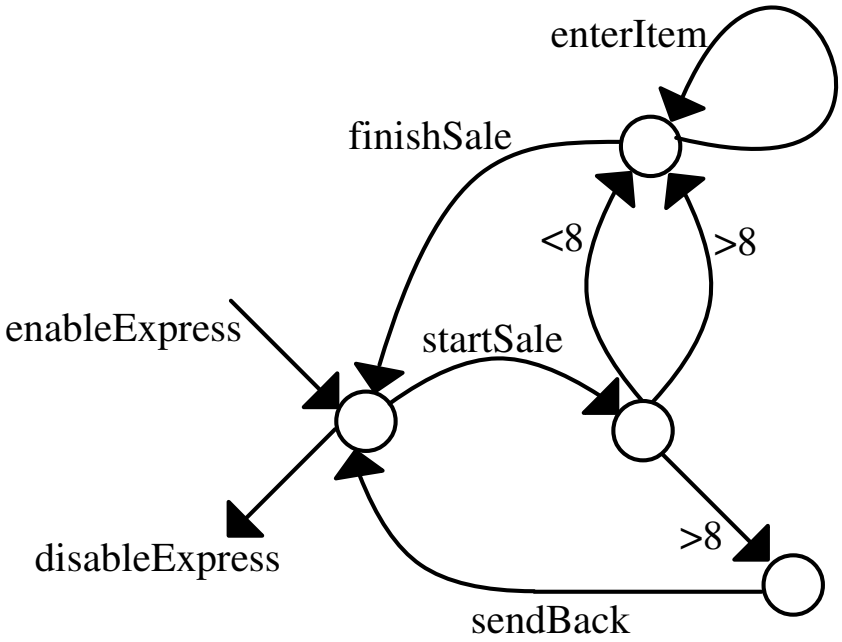
Specification of Prop2



- Permission to switch from normal to express mode
- Obligation to come back to normal mode
- Fairness constraint between normal and express mode

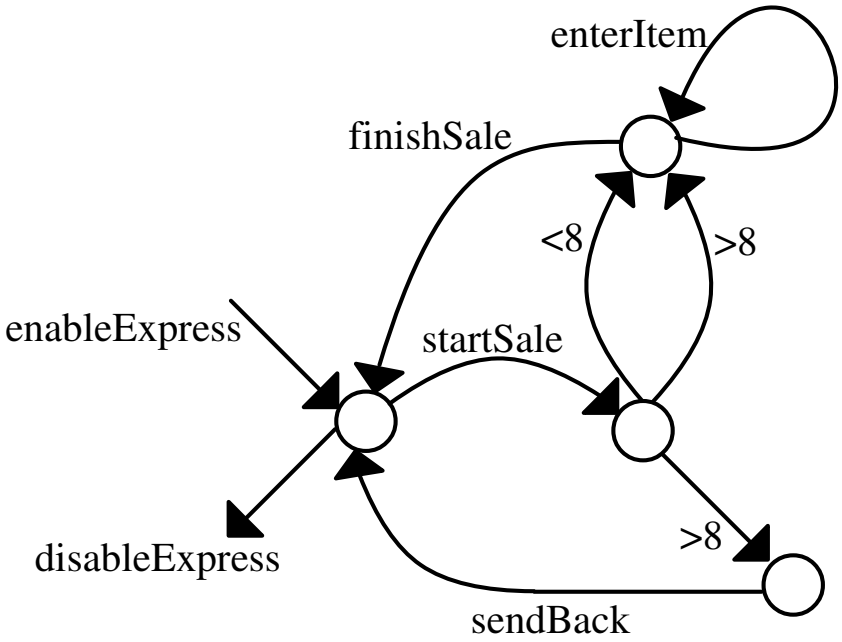
- Fairness left underspecified in CSP
- Cannot be described in CTL (fairness)
- Cannot be described in LTL (existential branch)
- Can be described in $\mathcal{CL}$ (modulo semantical treatment of fairness)

Specification of Prop3



- Obligation to serve customers with < 8 items
- Permission to service clients with > 8 items

Specification of Prop3



- Obligation to serve customers with < 8 items
- Permission to service clients with > 8 items

- Possible in CSP, but process not longer a refinement of original
- Described in CTL
- Cannot be described in LTL (existential branch)
- Can be described in $\mathcal{CL}$

Conclusions

	LTL	CTL	CSP	$\mathcal{CL}$
Prop1	✓	✓	✓	✓
Prop2	–	–	–	(✓)
Prop3	–	✓	(✓)	✓

- Different specification languages suitable to different purposes
  - Contracts only for normal behaviour: temporal logic would suffice
  - Composition and comparison of contracts: process calculi more flexibility
- Operational approach and temporal logic not very suitable to represent certain exceptional cases
  - $\mathcal{CL}$ could be encoded in CTL*
- Deontic approach suitable to represent obligations, permissions and prohibitions (and many exceptional cases)
  - Needs to be extended to capture more complex compensations (as present in *long transactions*)