Components, Objects, and Contracts

Olaf Owe and Gerardo Schneider and Martin Steffen

IfI UiO

3. September 2007
“A contract is a binding agreement between two or more persons that is enforceable by law.” [Webster online]
Contracts & e-contracts

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.

DEFINITIONS
4. 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 & e-contracts

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

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

• develop a notion of component model
• interface description by deontic contracts
• formal model for e-contracts
• formal semantics
• executable
• using Creol language
Creol: a concurrent object model

- executable oo modelling language **concurrent** objects
- formal semantics in **rewriting logics** /Maude
- strongly **typed**
- method invocations: synchronous or **asynchronous**
- recently: concurrent objects by (first-class) futures
- **dynamic reprogramming**: class definitions may *evolve at runtime*
Interfaces as types

- Object variables (pointers) are typed by interfaces (other variables are typed by data types)
- **Mutual dependency:** An interface may require a cointerface
  - Explicit keyword `caller`
  - Supports callbacks to the caller through the cointerface
  - Protocol-like behaviour
- Supports **strong typing:** no “method not understood” errors
- All object interaction is **controlled** by interfaces
  - *No explicit hiding* needed at the class level
  - Interfaces provide aspect-oriented specifications
  - A class may implement a number of interfaces
Contracts as behavioral interfaces
Contract specification language $\mathcal{CL}$

- formal specification language
- **expressive** enough to capture natural language contracts
  - contrary-to-duty (CTD)
  - contrary-to-permission (CTP)
- avoid certain **paradoxes** from deontic logic
A glimpse of $\mathcal{CL}$

\[\begin{align*}
\text{Contract} & := D \ ; \ C \\
C & := \phi \mid C_O \mid C_P \mid C_F \mid C \land C \mid [\alpha]C \mid \langle \alpha \rangle C \mid C U C \mid \Box C \mid \Diamond C \\
C_O & := O(\alpha) \mid C_O \lor C_O \\
C_P & := P(\alpha) \mid C_P \lor C_P \\
C_F & := F(\delta) \mid C_F \lor [\alpha]C_F
\end{align*}\]

- formal modal logic, combining aspects of
  - temporal,
  - deontic (O, P, F), and
  - dynamic logics
- formal semantics by translation into $\mu$-calculus $C_\mu$ variant
- model checking using nuSMV
- sophisticated action algebra
Conclusion & future work

- using Maude-engine for monitoring contracts
- conformance checking
- contracts-as-types

• FLACOS’07 – First Workshop on Formal Languages and Analysis of Contract-Oriented Software (in conjunction with NWPT’07): [http://www.ifi.uio.no/flacos07/](http://www.ifi.uio.no/flacos07/)
## References I

Creol: A type-safe object-oriented model for distributed concurrent systems.  

folk.uio.no/gerardo/nordunet3, 2007.

Components, objects, and contracts.  

Components, objects, and contracts.  
A short version appeared in the proceedings of SAVCBS’07.

A formal language for electronic language.  

Towards a formal definition of electronic contracts.  
Technical Report 348, Department of Informatics, University of Oslo, Oslo, Norway, Jan. 2007.