Verification of Fix Protocol Session Layer

Kamil Kuboň

AVoCS 2014
Agenda

- Motivation
- Project overview
- Fix Protocol Session layer
- QuickFix/J
- Process Algebra embedding in Isabelle/HOL
- Transformation from Java bytecode to Process Algebra embedding
- Verification
Motivation

- Case study for using theorem proving for basic properties of a communication protocol used in finance
- Enriching Isabelle/HOL with a theory code usable in such a study
Project overview

- Verification of Fix Protocol Session layer properties
- Target implementation is QuickFix/J framework implemented in Java

For this purpose following toolset will be developed:

- Conversion from Java bytecode to Isabelle/HOL
- Process algebraic language embedded in Isabelle/HOL
- Modal $\mu$-calculus embedding in Isabelle/HOL
Project diagram

Translation configuration

Specified and abstracted parts

QuickFix/J - core classes

Java Bytecode

Process algebraic language

Included in translation

Process algebraic representation of core protocol handling parts

Proofs

Isabelle/HOL

Modal μ calculus

Formulas to verify
Fix Protocol

▶ http://www.fixtradingcommunity.org/
▶ Communication protocol between counterparties in finance industry
▶ High volumes transferred
▶ Implemented and deployed on both “buy" and "sell" side of the market: banks, funds, corporate entities, retail products
▶ Support for wide range of financial products - foreign exchange, options, swaps..
Verification of Fix Protocol implementations

- acceptance testing and unit tests - part of QuickFix/J
- simulators
- companies providing certifications
- as per my knowledge no theorem proving and formalisation of properties in logic applied yet
Fix Protocol layers

▶ Session Layer
  ▶ tag=value content
  ▶ bidirectional stream of messages
  ▶ session handling - logon, logout, disconnection
  ▶ reliable delivery of messages
  ▶ assigning sequence numbers to messages
  ▶ resending messages
  ▶ handling duplicates, garbled messages
  ▶ order guarantee
  ▶ heartbeats

▶ Application Layer
  ▶ layer above the session layer
  ▶ contains a business logic
QuickFix/J

▶ http://www.quickfixj.org/

events from the network layer

SessionConnector

new messages

Session
+send(Message)

SessionState
+messageQueue
+getNextTargetMsgSeqNum()
+getNextSenderMsgSeqNum()

MessageStore

Application
+onCreate(SessionID sessionId)
+onLogon(SessionID)
+onLogout(SessionID)
+toAdmin(Message, SessionID)
+fromAdmin(Message, SessionID)
+toApp(Message)
+fromApp(Message)

Responder
accessing network layer

Session
+send(Message)

Responder
accessing network layer

SessionState
+messageQueue
+getNextTargetMsgSeqNum()
+getNextSenderMsgSeqNum()
Process Algebra embedding in Isabelle/HOL

- action call
- deadlock, successful termination
- non-determinism
- process call, recursion
- parallelism, communication
- summation
- method call semantics
- exception handling
- data types support
- heap representation
Purpose of the Process Algebra embedding

- specifying environment which should be included in the result
- specifying and abstracting the parts which are not being directly translated (interest in the core logic of the system being verifying, not in the complicated and efficient data structures)
- target of the bytecode translation
Transformation from Java bytecode to Process Algebra embedding

- bytecode instruction translated to a process call
- parameters of the process call should reflect stack and local variables manipulation
- instance variables modelled as processes with actions for modifying and retrieving data
- abstracted and specified parts should be included in the result
Verification

- Modal $\mu$-calculus embedding in Isabelle/HOL
- Verification of properties
  - Correctness of message recovery process
  - Messages being sent will be eventually delivered
  - Correctness of individual logon modes
  - Absence of deadlocks and race conditions
Expected contributions

- Verification of Fix Protocol Session layer using theorem prover
- Financial industry application
- Clarification of protocol properties formulating them in a formal logic
- Process algebraic language implemented in Isabelle/HOL, with translation from the bytecode
- Modal $\mu$-calculus embedding in Isabelle/HOL
Thank you