Digital Enforceable Contracts (DEC): Making Smart Contracts Smarter

Liu, L.; Sileno, G.; van Engers, T.M.

Publication date
2021

Citation for published version (APA):
**Background**

- Current smart contracts have limited capabilities of normative representations, making them distant from actual contracts.
- Normative contents (duty and power) can be modeled into logic-based representation.
- DEC provides a general architecture where various enforcement mechanisms are enabled by normative reasoning. For example, to check whether an action will lead to a duty.

**Actor-based Modular Architecture**

The architectural model is composed of a selected set of modules providing the functionality to run enforcement constructs.

**Actor** (the minimal unity of agency):
- Program - plan to achieve a given design goal
- Executor - internal control of the actor
- Message queue - communication channel
- Monitor - listeners that hook to events or facts
- Monitor manager - handle monitors
- Regulator - normative reasoning

**Example: A Data-sharing Scenario with GDPR**

1) John (data-subject) attempts to revoke his consent of using his data from Bank (data-controller).
2) The executor sends query to the regulator to check related permissions and duties. (According to GDPR, Bank, as data-controller, has the duty to fulfill this request.)
3) The executor sends this request to the queue.
4) The request is then sent to Bank.
5) The executor asks monitor manager to create a monitor to check for violation.
6) A monitor is created.
7) The monitor checks messages from Bank with a timeout mechanism.
8) When the duty is due and not fulfilled, the monitor will be aware of the violation.
9) The monitor reports the violation.
10) Monitor manager notifies the executor of the violation.
11) The executor takes actions to deal with the violation.

**Acknowledgements**

This research is funded by the Dutch Organization for Scientific Research (NWO) under contracts 638.009.014 (SSPDDP project) and 638.001.001 (DL4LD project).

**References**