

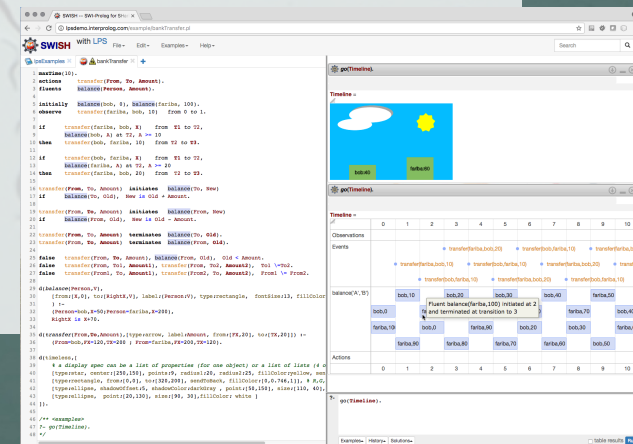
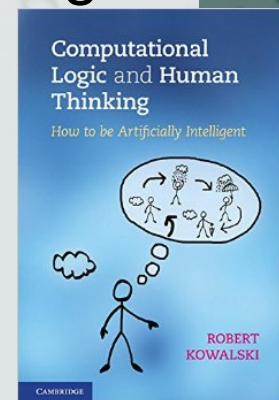
Logical Contracts

Logical Contract Server Preview
RuleML Webinar, Nov 24, 2017
Miguel Calejo (*with LC team*)

Preamble: Logical Production Systems (LPS)

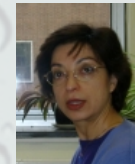
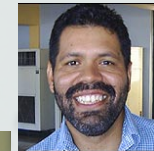
- Research by Kowalski and Sadri at Imperial College
 - Legal Reasoning, British National Act, Obligation as goal satisfaction, etc.
 - *Computational Logic for Human Thinking* book
 - Logical agents with Dávila
 - *Computational Logic for Use in Teaching* with Calejo
 - Several LPS implementations

- <http://lps.doc.ic.ac.uk>



Logical Contracts?

- [Miguel Calejo](#), CTO
- [Bob Kowalski](#), Chief Scientist
- [Jacinto Dávila](#), Senior Engineer
- [Fariba Sadri](#), External Research
- [Alex Garcia](#), Business Dev



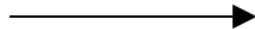
imperial
innovations

- Imperial College London spinoff (*now courting investors*)
- Enhance and apply LPS to...
 - Smart contracts: a good fit

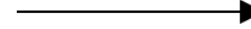
Sales pitch du jour...



**Enterprise
Systems**



**Logical
Contracts**



**Simplified Smart
Contracts Solution**

Blockchain

Integrating enterprise systems with smart contracts
Imperative logic in plain English

Vision through example

- *Rock-Paper-Scissors gambling game*
- *1 click:*
 - *simulation is displayed, graphical and narrative*
- *1 click:*
 - *contract executing on cloud with new Ethereum address*
- *We're done:*
 - *bets are received, game is decided, prize paid*
 - *Ethereum retains history and contract reference*

RSP gambling (simulated)

Logical Contracts -- Alpha Dev x

Not Secure demo.logicalcontracts.com/example/RockPaperScissorsBase.pl

Logical Contracts
File Edit Examples Help

RockPaperScissorsBase fintechExamples

```

3
4 beats(scissors, paper).
5 beats(paper, rock).
6 beats(rock, scissors).
7
8 events transaction_from(_From,_Input,_Ammount).
9 fluents played(_Player,_Choice),reward(_Total), gameOver.
10 actions pay(_Player,_Prize).
11
12 initially reward(0).
13
14 % simulate input events:
15 observe transaction_from(miguel,rock,1000) from 1 to 2.
16 observe transaction_from(bob,paper,1000) from 1 to 2.
17 observe transaction_from(alex,paper,1000) from 2 to 3. % one player too many!
18
19 transaction_from(From,Input,Wei) initiates played(From,Input) if
20     Wei>0, not played(From,_).
21
22 num_players(N) at T if
23     findall(P, played(P,_) at T, L), length(L,N).
24
25 false num_players(N), N>2.
26
27 transaction_from(_Player,_,X) updates Old to New in reward(Old) if
28     New is Old+X.
29
30 pay(_,Prize) updates Old to New in reward(Old) if New is Old-Prize.
31
32 if played(P0,Choice0) at T1, played(P1,Choice1) at T1, P0\==P1, beats(Choice0,Choice1), not gameOver at T1
33 then initiate gameOver from T1, reward(Prize) at T1, pay(P0,Prize) from T1 to T2.
34
35 if played(P0,Choice) at T1, played(P1,Choice) at T1, P0 @> P1, not gameOver at T1
36 then initiate gameOver from T1, reward(Prize) at T1, Half is Prize/2, pay(P0,Half) from T1, pay(P1,Half) from T1.

```

go(Timeline).

Rejected observations [transaction_from(alex,paper,1000)] attempting to satisfy prospective preconditions

Timeline =

	-2	-1	0	1	2	3	4	5	6	7
Observations										
Events						pay(bob,2000)				
						initiate gameOver				
						transaction_from(bob,paper,1000)				
						transaction_from(miguel,rock,1000)				
gameOver						gameOver				
played('A','B')						miguel,rock				
						bob,paper				
reward('A')			0			0				
					200					
Actions										

0.125 seconds cpu time

go(Timeline).

Examples History Solutions

☐ table results Run

LPS for logic programmers:

- LPS is a Prolog superset, adding explicit time...
 - “Time” as a sequence of discrete cycles
- Literals can be **timeless** as usual... or not:
 - **Fluents**: true over a cycle
 - **Events/actions**: happen in cycle transitions
- Extra syntax:
 - Fluent and event rules and declarations, external observations
 - Post conditions (actions changing fluents), integrity constraints
 - Reactive rules
- A Prolog program executes “instantly”. But a LPS program executes over time cycles:
 - Reactive rules introduce *parallel* (AND) goals
 - Fluents and events/actions mean.. delay

Timeless truth:

```
beats(scissors,paper)
```



Action!

Event:

```
happens(transaction_from(bob,paper,1000),1,2)
```

Action:

```
pay(bob,2000) from 2 to 3
```

Fluents:

```
holds(reward(2000),2)
```

```
gameOver at 3
```

Blockchain 101

- “Global” database, or “distributed ledger”
 - Grows monotonically with new versions (“blocks”) every few seconds
 - Contains account balances and transactions
- Ethereum, Hyperledger, ...
 - Can also contain and execute code (smart contracts): Solidity, Javascript
- Ethereum interface for LPS
 - LPS in cloud off blockchain; flexibility, abstraction from blockchain du jour
 - `e_getBalance(+Account,+Block,-Value)`
 - `e_transaction(+Block,?From,?Input,?Wei,?To)`
 - `e_sendTransactionWithAtom(+From,+To,+Value,+Message,-Tx)`
 - `e_existsTransactionReceipt(+Tx)`
 - For anything else doable with [Ethereum RPC](#):
 - `e(method(Argument1,Argument2,..Result)`

RSP on Ethereum blockchain

```
1 % Rock, Paper, Scissors gambling on Ethereum!
2 % or in hexadecimal: '0x726f636b', '0x7061706572', '0x7363697373667273'
3 maxRealTime(120). % 2 minutes game lifetime
4 beats(scissors, paper).
5 beats(paper, rock).
6 beats(rock, scissors).
7
8 prolog_events e_transaction(latest, _From, _Input, _Wei, _To). % Generate events from the blockchain
9
10 e_transaction(latest, From, Input, Wei, To) initiates played(From, Input) if
11     lps_my_account(To), Wei > 0, not played(From, _).
12
13 fluents played(_Player, _Choice), gameOver.
14
15 reward(R) at T if % intensional fluent obtained from the blockchain
16     lps_my_account(A), e_getBalance(A, latest, V) at T,
17     R is round(V*0.9). % keep 10% for gas
18
19 num_players(N) at T if
20     findall(P, played(P, _) at T, L), length(L, N).
21
22 false num_players(N), N > 2.
23
24 pay(Player, Prize) from T1 to T3 if % plan / macro action on the blockchain
25     lps_my_account(Us),
26     e_sendTransaction(Us, Player, Prize, PaymentTx) from T1 to T2,
27     e_existsTransactionReceipt(PaymentTx) at T3.
28
29 if played(P0, Choice0) at T1, played(P1, Choice1) at T1, P0 \== P1, beats(Choice0, Choice1), not gameOver at T1
30 then initiate gameOver from T1, reward(Prize) at T1, pay(P0, Prize) from T1 to T2.
31
32 if played(P0, Choice) at T1, played(P1, Choice) at T1, P0 @> P1, not gameOver at T1
33 then initiate gameOver from T1, reward(Prize) at T1, Half is Prize/2, pay(P0, Half) from T1, pay(P1, Half) from T1.
```

RSP on Ethereum blockchain

Logical
Contracts

The screenshot shows the Logical Contracts web application. The left pane contains a code editor with the following Solidity-like contract code:

```
1  en{ "  
2  
3  the maximum time is 120.  
4  
5  the prolog events are:  
6    e_transaction a time a first address an input an amount a second address.  
7  
8  the fluents are:  
9    the game is over, known as gameOver,  
10   a player has played a choice, known as played.  
11  
12  scissors beats paper.  
13  paper beats rock.  
14  rock beats scissors.  
15  
16  When e_transaction latest a first address an input an amount a second address  
17  and lps_my_account the second address  
18  and the amount is greater than 0  
19  and it is not the case that the first address has played a second choice  
20  then the first address has played the input.  
21  
22  the reward is a number, known as reward, at a time if  
23  lps_my_account an address  
24  and e_getBalance the address latest a value at the time  
25  and the number is approximately 0.9 times the value.  
26  
27  the players are a number, known as num_players, at a time if  
28  the number at the time is  
29  the sum of all  
30  a player has played a value at the time.  
31  
32  It must not be true that  
33  the players are a number, known as num_players, at a time  
34  and the number is greater than 2.  
35  
36  a player pays a prize, known as pay, from a first time to a second time if
```

The right pane shows a visual state machine diagram with states like 'played(A,B)', 'reward(A)', and 'Actions'. Below the diagram is a console window showing the execution log for the contract.

The screenshot shows the Etherscan Ropsten (Revival) Testnet interface. The address `0xf0b986c9bb9b7d0d9754c7052d8b07b57FA70a6` is selected. The account has an ETH balance of 0.2 Ether and 2 transactions.

Transactions

TxHash	Block	Age	From	To	Value
0x62b280d51f8434...	(pending)	23 secs ago	0xf0b986c9bb9b7d0d9754c7052d8b07b57FA70a6	0xd9d046661b63e0...	0.09 Ether
0x98fd64e114f6566...	(pending)	23 secs ago	0xf0b986c9bb9b7d0d9754c7052d8b07b57FA70a6	0x643919713fca23...	0.09 Ether
0x2b99455306024c...	2016949	52 secs ago	0x643919713fca23...	0xf0b986c9bb9b7d0d9754c7052d8b07b57FA70a6	0.1 Ether
0xf1b5bb08dd71dd...	2016949	52 secs ago	0xd9d046661b63e0...	0xf0b986c9bb9b7d0d9754c7052d8b07b57FA70a6	0.1 Ether

Demo link at <http://logicalcontracts.com/server/>

Conclusion

- Demo and more info at <http://logicalcontracts.com/server>
 - LPS multi contract server, implemented with SWI Prolog and SWISH
 - Web editor, visualizations
 - Preliminary formal English, explainer
 - Web services (event injection, remote actions)
 - Hibernation (a contract can suspend and resume in another engine)
 - Ethereum logical API (via a local geth node)
 - Demo playground with Ethereum testnet accounts
- Upcoming
 - Pilot projects, to validate LPS for smart contracts
 - Formal English improvements, other languages
- Open source pledge😊
 - ...except some dev tools and external interfaces

THANKS! mc@logicalcontracts.com