First Monday, Volume 22, Number 3 - 6 March 2017

PEER-REVIEWED JOURNAL ON THE INTERNET

Towards a post-cash society: An application to convert fiat money into a cryptocurrency

by Steve Huckle, Martin White, and Rituparna Bhattacharya

Abstract

In this paper, we discuss an application that uses blockchain technology to transfer fiat money into a cryptocurrency — Ether. A typical use of this technology could be to become a component of a larger system, whereby, after traveling, a customer can exchange leftover foreign currency for their local denomination. However, a more interesting application could be to convert fiat money into a cryptocurrency to facilitate a *demonetisation* scheme, such as that implemented recently in India. In the latter context, we describe the development of our blockchain application against the ramifications of *demonetisation* and whether the Indian government could have augmented that scheme through technology such as ours. We discuss why the Indian government has not contemplated such a measure, which also leads to a discussion of whether they might have considered adopting their own cryptocurrency. However, even though the Indian public seems willing to adopt the technology, we find that unlikely. Finally, we show that our application demonstrates that fiat money to cryptocurrency conversion is technically feasible, but the Indian government is unlikely to consider such technology due to issues surrounding monetary sovereignty.

Contents

Introduction

This paper introduces a prototype blockchain technology that could become the basis of an application for converting physical fiat money, which is a currency established as money through government regulation [1], into Ether [2], the form of cryptocurrency used by the Ethereum blockchain [3]. The code was the result of a scenario called "John's International Tour", introduced in the paper "Internet of Things, blockchain and shared economy applications" [4], where, after a trip abroad, John uses a mobile blockchain-based peer-to-peer currency exchange application to swap his leftover foreign currency for local money. Hence, the intention was to create an application that did not require the direct intervention of a central authority, or organisation, to exchange one form of fiat money for another. However, during development of our currency conversion tool, the Indian government announced their intention to demonetise their 500 and 1,000 Rupee notes [5], a move that would, reportedly, take out of circulation more than 80 percent of their physical cash [6]. When the news broke, our currency exchanger was already capable of converting physical cash into Ether, but India's removal of their banknotes gave us an "Aha!" moment because we realised the huge implications of the act of turning fiat money into its cryptocurrency equivalent. Given that, we wondered whether our prototype application could be exploited in a demonetisation scheme whereby a sovereign state transitions to a post-cash society through a digital application that converts fiat money to a digital cryptocurrency. This paper, then, considers that possibility. First, we give an overview of our prototype code. Then we discuss demonetisation and India&sdquo;s evolution of their digital financial services. Finally, we examine

whether the Indian government could have used a version of our application for turning their discontinued notes into cryptocurrencies..

Below we give a description of the code.

A prototype blockchain-based currency conversion application

Using the terminology of lean software development [7], the blockchain application we will describe is a minimally viable product (MVP) [8]. That translates into the idea that the code is not ready for a production environment, but it has enough functionality to demonstrate the ideas presented in this paper. Furthermore, we do not intend to list every line of the application or even some essential functions (such as displaying the contract's current balance of Ether), but rather, we shall show the key code that demonstrates the application's 'post-cash' aspects. However, should more detail be required, the code is stored in a private GitHub repository, and access to that is available on request.

Architectural overview

The application consists of the following architecture:

1. An Ethereum blockchain [3].
2. A smart contract, written in the language *Solidity* [9], which handles the exchange of physical currency into Ether. A smart contract is an addressable script that represents verifiable application logic on the blockchain [10].
3. A JavaScript front-end, which we used to create an administrator interface.
4. Another JavaScript front-end for the application itself, which would allow users to exchange their physical cash for Ether.

The development environment

The development of the currency exchange application took place on a machine running Ubuntu Xenial Xerus [11]. We installed geth [12], which is the command-line interface for running a full Ethereum node and we used the Embark Framework to run that node and ease deployment of our smart contracts [13]. React was the frontend JavaScript framework we used [14]. That requires NodeJS on the back-end [15].

The currency exchange smart contract

The smart contract that handles the currency exchange needs funding with Ether so that it can seamlessly manage the conversion of fiat currency. Figure 1 shows the execution flow of that funding.

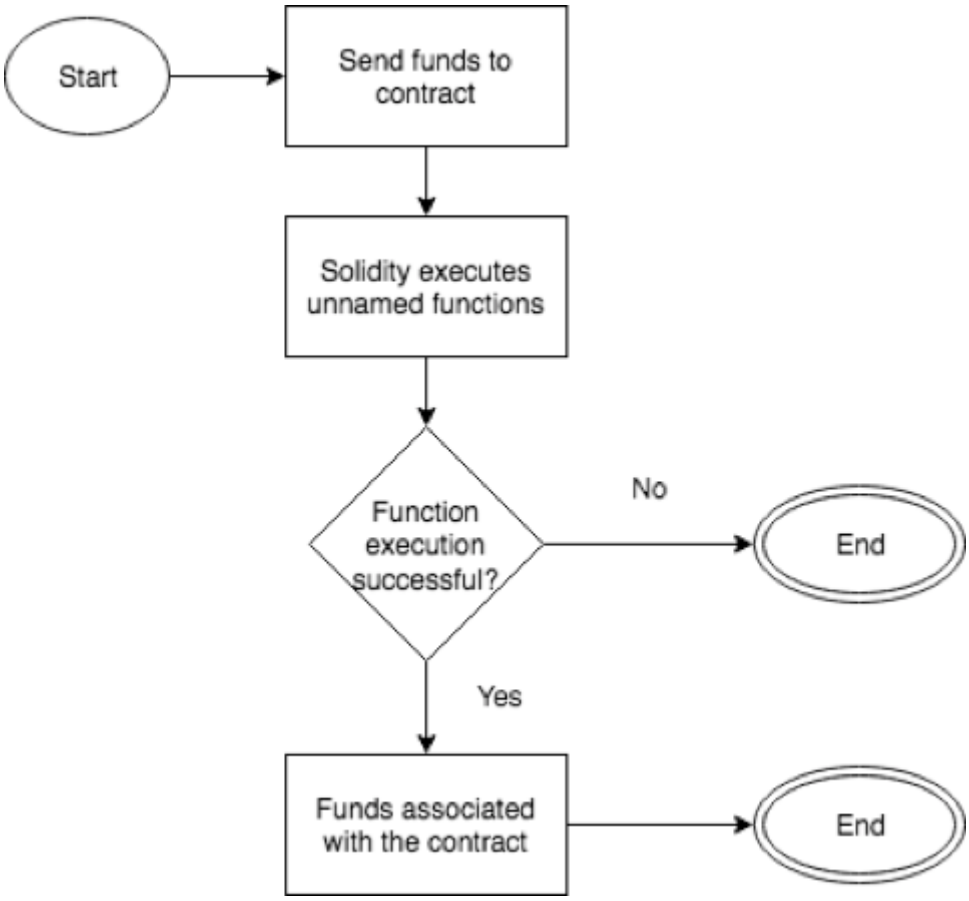

Figure 1: Sending funds to an Ethereum smart contract.

If, when calling a contract, none of the other functions match the function identifier, Solidity automatically executes any defined unnamed function [16]. After that, if the function runs successfully, any Ether sent will be associated with that contract. The unnamed function must not have arguments and cannot return anything. Furthermore, the amount of work it is allowed to do is limited; Ethereum introduces the concept of *gas*, which is the internal pricing used for running code [17]. An unnamed function cannot exceed 2300 gas [16], which effectively means it cannot write to the blockchain or perform any complex calculations. Below is the unnamed function used by the currency exchange smart contract. All it does is trigger a defined *Funded* event that the frontend captures and acts upon accordingly. The event takes two parameters, 1) the address from which the unnamed function was called and, 2) the amount of Ether sent. Finally, the function modifier *payable* ensures that the unnamed function can receive Ether.

```
event Funded(address sender, uint rate);

function() payable {
    Funded(msg.sender, msg.value);
}
```

The rate of exchange of fiat money for Ether is set in the smart contract function shown below. It takes three parameters, 1) the currency code (a three letter string corresponding to the ISO 4217 currency code standard [18]), 2) the Ether exchange rate for that currency, and 3) the time of the setting of the rate. The function uses the *sha3* hash function [19] to produce a 32-byte value from the currency code, which is used as the key to a hashtable that stores the rate.

```
mapping(bytes32 => uint256) private rates;

function setRate(string _code, uint256 _rate) public {
    var key = sha3(_code);
    rates[key] = _rate;
}
```

Next, we show the functions responsible for exchanging physical cash for Ether; *placeOrder* and *completeOrder*. The 'place' and 'complete' workflow mimics the behaviour of the cryptocurrency exchange platform Kraken [20]. Currency exchange orders are stored in a Struct, and they can be either *open*, *deleted* or *completed*. A hashtable stores the structs, which are keyed on the time and address of whom placed the order. Orders contain the type and amount of physical currency, as well as the amount of Ether to which the physical currency converts (which the callee ascertains by calling the *getRate* function). Successfully placed orders will trigger another event that the frontend captures.

```
enum OrderStatus { OPEN, DELETED, COMPLETED }

struct Order {
    address creator;
    string offerCurrency;
    uint256 offerAmount;
    uint256 etherAmount;
    OrderStatus status;
}

mapping(bytes32 => Order) private orders;

event OrderPlaced(uint256 _epochTime, address _creator);

function getRate(string _code) public constant returns (uint256) {
    var key = sha3(_code);
    return rates[key];
}

function placeOrder(uint256 _epochTime, address _creator, string
_offerCurrency, uint256 _offerAmount, uint256 _etherValue) public {
    var key = getOrderId(_epochTime,_creator);
    orders[key] =
Order(_creator,_offerCurrency,_offerAmount,_etherValue,OrderStatus.OPEN);
    OrderPlaced(_epochTime, _creator);
}
```

Finally, the function below shows the process of completing an order. The details of the physical cash deposited are stored in a hash table, and the equivalent amount of Ether is sent to the address that created the order. If the order completes successfully, two events are triggered, namely, an *OrderCompleted* event and a *Deposited* event. Otherwise, the order is kept open, and the physical cash is returned.

```
mapping(bytes32 => uint256) private deposits;

event OrderCompleted(uint256 _epochTime, address _creator);
event Deposited(string _offerCurrency, uint256 _offerAmount);

function completeOrder(uint256 _epochTime, address _creator, string
_offerCurrency, uint256 _offerAmount, uint256 _etherValue) public {
    var orderKey = sha3(_epochTime,_creator);
    Order thisOrder = orders[orderKey];
    if ( thisOrder.status == OrderStatus.OPEN ) {
        thisOrder.status = OrderStatus.COMPLETED;
```

```
        var depositKey = sha3(_code);
        deposits[depositKey] = _offerAmount;
      if (_creator.send(_etherValue)) {
        OrderCompleted(_epochTime,_creator);
        Deposited(_offerCurrency,_offerAmount);
      } else {
        thisOrder.status = OrderStatus.OPEN;
        uint256 depositedAmount = deposits[depositKey];
          uint256 newAmount = depositedAmount - _offerAmount;
          deposits[depositKey] = newAmount;
      }
    }
}
```

Why use a hashing function?

In the functions *setRate* and *completeOrder*, shown above, the hashing function *sha3* produces a key from a three-letter currency code. Obviously, it would be more computationally efficient to use the currency code itself as the key. However, the code shown is a simplification of that written; in reality, we handle storage via a separate class deployed to the blockchain. That requires a 32-byte key for its hash tables, hence our use of *sha3*. Furthermore, because, potentially, many applications could use that storage class, we need to use a unique key. So rather than the code shown, we use *sha3* with a timestamp as well as the currency code. Below we show a call to that storage class.

```
ExternalStorage private storageContract;
uint private timeStamp;

function ExchangeRate() {
    storageContract = new Storage();
    timeStamp = now;
}

function setRate(string _code, uint256 _rate) public {
    var key = sha3(_code, timeStamp);
    storageContract.setUInt256Value(key,_rate);
}
```

Here's a snippet of the Storage class.

```
class Storage {

    mapping(bytes32 => uint256) private uInt256Storage;

    function getUInt256Value(bytes32 record) public constant returns
(uint256) {
        return uInt256Storage[record];
    }
    function setUInt256Value(bytes32 record, uint256 value) public {
        uInt256Storage[record] = value;
    }

}
```

The yellow paper on Ethereum states that the word size of the Ethereum virtual machine is 256-bit [21]. That figure was specifically chosen to facilitate the *sha3* hashing scheme. Furthermore, the total gas payable for memory usage is proportional to the smallest multiple of 32 bytes. Hence, 32-byte values are the most efficient means of storage within Ethereum, further justifying our use of *sha3* as a means of producing keys for our hash tables.

It would be fair to wonder why we used unsigned integers for the amounts to exchange? Wouldn't it be easier to use fixed point decimals to represent values such as £3.52? Unfortunately, at the time of writing, Solidity does not support fixed-point arithmetic. However, circumventing that is simply a matter of using units that are much smaller than any fixed point number used. Ethereum's smallest denomination is the Wei [22] — one Ether is 1,000,000,000,000,000,000 Wei. So, when the front-end calls completeOrder and passes in the physical cash's *offerAmount*, it first has to convert the amount to Wei. Similarly, when the front-end calls *getRate*, the rate it receives is in Wei.

Front-end interfaces

The front-end interface to our smart contract uses the *web3.js* JavaScript library, which exposes an application programming interface (API) containing the code for interacting with Ethereum [23]. The general pattern for using the library is to instantiate a *web3.js* object, then set the location of the running blockchain and then specify the default Ethereum account. To use the smart contract, define the contract's application binary interface, then register that interface with *web3.js* and detail the contract's address on the blockchain. The final step is to create an object that uses the contract at that address, which we show below because we the variable assigned the contract address in all the proceeding examples. The Ethereum Wiki gives some further examples of the steps required to use *web.js* [24].

```
const exchanger = exchangerContract.at(contractAddress)
```

The application has both user and administrator interfaces, which we discuss next.

The administrator interface

The administrator interface of the application provides two primary functions:

1. Funding of the smart contract responsible for exchanging fiat money into Ether.
2. Setting exchange rates.

Funding the smart contract involves calling our smart contract's unnamed function.

```
web3.eth.sendTransaction({from: adminAccount, to: contractAddress, value:
web3.toWei(funds,"ether")})
```

Setting the exchange rates means a call to the smart contract's *setRate* function.

```
exchanger.setRate(currency, web3.toWei(rate,"ether"))
```

The *setRate* function cannot return a boolean to confirm that the rate was set successfully because the function updates the blockchain, which creates a transaction that must be proved [25] and that proof does not happen instantaneously. Therefore, rather than returning a value detailing success, a transaction hash is returned, which is then used to monitor how the transaction progresses. Below, we show how the front-end watches for a *RateSet* event. Later, we will briefly describe using the transaction hash returned.

```
var currencyCode = ""
var rate = 0
var rateSet = exchanger.RateSet(function(error, result) {
    if (!error) {
        currencyCode = result.args.code
        rate = web3.fromWei(result.args.rate.toNumber())
    } else {
        console.error(result)
    }
})
```

The user interface

The user interface has one primary function, shown in Figure 2, namely, to exchange fiat money for Ether (at the set rate). The first step is to get the exchange rate for the currency. Because the smart contract's *getRate* function does not update the blockchain, it can return that rate directly.

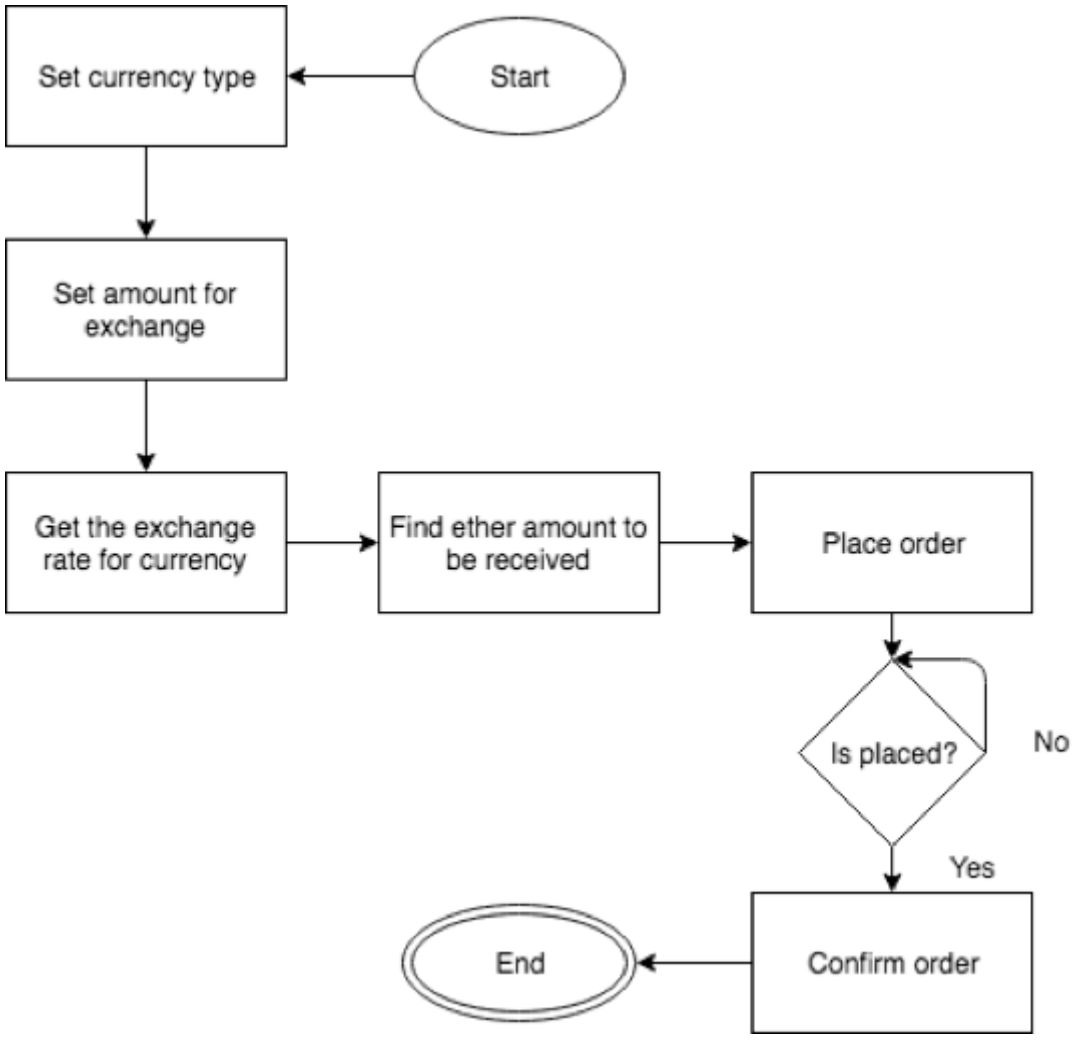

Figure 2: Exchanging fiat money for Ether.

```
const rate = exchanger.getRate(currency).toNumber()
```

The next step is to place an order. The caller must send the time of placing the order, the address of the Ethereum account to which to transfer the exchanged Ether, the type of fiat money for exchange and its amount, as well as the amount of Ether they should receive (based on the exchange rate):

```
const epochTime = (new Date).getTime()
const transactionHash = exchanger.placeOrder(epochTime, account, currency,
web3.toWei(physicalAmount,"ether"), web3.toWei(etherAmount,"ether") )
```

Note that *placeOrder* returns a transaction hash that, as explained above, can be used to monitor how the transaction progresses. The application itself uses the transaction hash to set a boolean that confirms placement of the order. However, the code shown displays the general principles well enough:

```
web3.eth.filter('latest').watch(function (error, result) {
    if (error) {
        console.error(error)
    } else {
        const block = web3.eth.getBlock(result, true)
        let transactions = block.transactions
        for(let i = 0; i < transactions.length; i++)
        {
            if( transactionHash == transactions[i].hash ){
                console.log("Order Placed!"})
            break"Putting aside that India has demonetised before in 1946 and
1978 so you have to ask the question why?
            }
        }
    }
})
```

Finally, the order must be confirmed.

```
const transactionHash = exchanger.completeOrder(epochTime, account, currency,
web3.toWei(physicalAmount,"ether"), web3.toWei(etherAmount,"ether"))
```

Running the application in a production environment

Currently, to conduct an exchange using our application, the user interacts with a Web-based interface and simply selects the currency to convert from a drop-down list. Then they input the amount of cash to be converted. Although that is not a real exchange, it is not difficult to envisage how the process might involve physical currencies. For instance, the same input process could take place via some electronic currency reader similar to that used every day in vending machines. Then there's the kiosk used by the company Fourex [26], which allows users to exchange a range of foreign currencies for U.S. dollars, British pounds, or the Euro. Hence, Fourex kiosks are already capable of recognising fiat money (coins and banknotes) of different types and converting them to other denominations. Our proposal is, then, that Fourex uses a production version of our application to extend their kiosks so that they are capable of converting foreign currencies into a cryptocurrency. We will not get too involved in a detailed description of such an application, but we envisage the kiosk as somehow scanning an Ethereum wallet's address on a user's smartphone, to which they transfer exchanged Ether. Furthermore, rather than an administrator inputting exchange rates, they could be fetched automatically via a cryptocurrency exchange rate API, such as Poloniex [27]. Moreover, it does not take too much imagination to see how our MVP could work for real and how the Indian government could use, for example, Fourex kiosks to obfuscate their larger denomination notes. Quite simply, the notes could be deposited in the kiosks, exchanged for Ether, and then destroyed.

The code we have shown above is, essentially, very simple. Hence, there are no technical issues to converting fiat money into cryptocurrencies. There are other complications, though, and that is a discussion to which we turn to next.

Demonetisation

On 8 November 2016, at 8 pm Indian Standard Time, the Prime Minister of India, Mr. Narendra Modi, announced the *demonetisation* of all 500 and 1,000 Rupee banknotes in a live televised address; come midnight on that day, those notes would become invalid. Below, we discuss the scheme, its ramifications [28], and wonder whether our application could have aided the process.

Sovereignty

Sovereign is a late thirteenth, early fourteenth-century word derived from the old French *soverain*, meaning "highest, supreme or chief." That comes from the Vulgar Latin *superanus*, meaning "chief or principal," which itself is a derivation from the Latin *super*, meaning "over" [29]. Sovereignty, then, means *supreme power*. Thus, a sovereign is the ultimate overseer of a country's decision-making processes [30] and in modern law, sovereignty refers to the full right and power of a nation state to govern itself. Modern democratic nations have deferred monetary sovereignty to their central banking authorities [[31]. In India, a 1934 Act conveyed that privilege on the Reserve Bank of India (RBI) [32]. Like other central banks around the world, the RBI is responsible for the overall supply of money, and thus, the control of the country's inflation rate. That helps ensure financial stability, thus increasing the level of trust the public has for their currency. That relationship between trust and its issuing authority, which money creates, is significant [33].

When *demonetisation* is put in the context of public trust, India's move seems surprising. Indeed, there have been significant protests against the move [34], to which the Indian government reacted by relaxing some of the conditions behind the demonetisation [35]. Indeed, Mr. Modi tried to reassure the public that the scheme was "not born from arrogance," but rather it was a measure meant to tackle corruption and tax evasion and to drive "black money out of the shadows" [36]. He acknowledged the pain that ordinary Indians had experienced due to the long queues at the nation's banks [37] where people could, until the end of December, deposit their 500 and 1,000 Rupee notes and thereby subject their unaccounted wealth to taxation [38].

The informal sector

The World Bank describes the *informal sector* as that which leads to unofficial earning strategies. That might be through casual and temporary jobs, or businesses activities such as tax evasion and avoidance of other government or institutional regulations, as well as underground activities, such as crime and corruption and, crucially: "activities not registered by statistical offices" [39]. A 2014 study showed that India's *informal sector* amounts to some 40 percent of the Indian economy [40]. As a consequence, the tax income of the Indian government is just one-third of that which they are due [40].

Furthermore, a September 2016 report by McKinsey declared that: "Two billion individuals and 200 million micro, small, and midsize businesses in emerging economies today lack access to savings and credit" [41]. The report argues digital finance could have an enormous impact on the 'developing' world, by boosting growth to the value of US$3.7 trillion, which, as Figure 3. shows, equates to a six percent worldwide GDP growth by 2025. Their findings were given further credence by the Global Innovation Exchange, who argue that cash is an inefficient medium of exchange and its widespread use costs the Indian economy $3.5 billion annually. They conclude that the Indian Government would save one percent of their annual economic output by digitising physical cash.

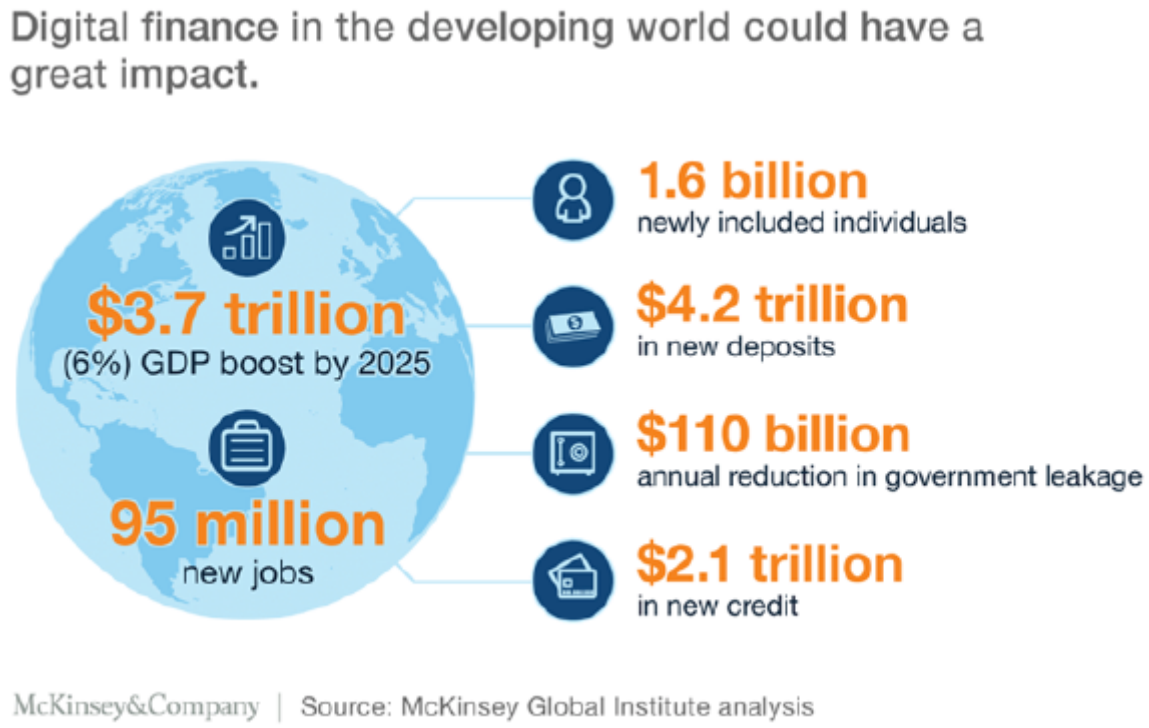

Figure 3: Digitising the developing world [41].

Digitising the Indian economy

The Indian government's demonetisation scheme is not the first time banknotes have been removed from circulation. In early 2014, the RBI announced that all pre-2005 bank notes would be withdrawn [42]. Furthermore, it is the latest in a raft of measures aimed at digitising the Indian financial system. In

September 2010, the Unique Identification Authority of India starting issuing 12-digit unique identification numbers, termed as Aadhaar, to all of its citizens [43]. The numbers link to a resident's biometric data, such as their photographs, fingerprints and iris scans. RuPay, a portmanteau of the words Rupee and payment [44], was launched in 2011; it is a multilateral domestic payment card scheme that is accepted at all Indian automatic telling machines (ATMs) and which offers a viable alternative to MasterCard and Visa [45]. In July 2015, Mr. Modi announced *Digital India* [46], a scheme to help improve Internet infrastructure, online government services and computer literacy. April 2016 saw the launch of the Unified Payments Interface, a system where participating banks unify all of their functionality into a single mobile application. The end of 2016 saw a new initiative to replace debit and credit cards with Aadhaar-enabled payments [47].

However, by far the most wide-ranging digital finance scheme was launched in August 2014, when Mr. Modi announced "Pradhan Mantri Jan Dhan Yojana," the "Prime Minister's People Money Scheme," an initiative for providing the Indian population with universal access to banking facilities [48]. The scheme comes with a range of benefits, including the ability to open a bank account with no deposit. Those accounts come with free life insurance, loan benefits and mobile banking facilities [49]. There were also some incentives for digital payments, including discounts on fuel, insurance policies, railway tickets and toll roads. In a further bid to encourage the use of the banking system, the government issued credit cards to farmers and digital point of sale devices were distributed to Indian villages [50].

Pradhan Mantri Jan Dhan Yojana seems well justified. In 2011, just 35 percent of the adult Indian population had a bank account, and although that rose to 53 percent by 2014 (equating to an increase of 175 million people) [51], it still does not compare favourably to the global average of 62 percent [51]. However, by the end of January 2015, matters continued to improve, and a further 125 million bank accounts were opened [51]. However, there is still a way to go; during 2015, only 11 percent of the Indian public used a debit card, and 97 percent of Indian retail transactions were conducted in cash [52]. Indeed, by November 2016, 68 percent of total Indian transactions were cash based. Furthermore, in 2014, 43 percent of India's accounts were completely inactive [51]. Unsurprisingly, immediately after the announcement of *demonitisation*, there was a large spike of deposits [53], a tenfold increase in digital transactions [54] and an overall 268 percent year-on-year increase in collected taxes [55]. Furthermore, India has quickly become the second largest smartphone market in the world, with 220 million users [56]. That is helped create a fast adoption of electronic wallets, capable of more than settling phone contracts or paying bills, but also online shopping [57].

Cryptocurrency adoption

Currently, it does not appear that the Indian government is looking at a push towards wide scale adoption of cryptocurrencies; the RBI have remained almost silent on the subject [58]. One reason for that could be a topic discussed earlier — sovereignty; even the ancient Chinese recognised the importance of controlling money supply: "Whoever wished to remain in power and see his domain well governed should jealously guard the management of the monetary standard and the monopoly of issuance" [59]. By legislating in favour of cryptocurrencies, the Indian government would, effectively, be giving up their ability to maintain ultimate control of their economy. Indeed, perhaps for that same reason, very few governments recognise any form of cryptocurrency as legal tender. In fact, it's far more common for countries to actively legislate against them [60].

However, could India create its own cryptocurrency? Ethereum offers the ability to implement *private blockchains* [61], which are centralised (currency) systems with an infusion of cryptographic auditability [[62]. In other words, such systems could resemble a digital equivalent of the monetary control that the Indian government has conferred on the RBI. Indeed, there appears to be some take-up of blockchains used for that very purpose. An especially interesting example is the U.K. government's Royal Mint [63], who are investigating "Royal Mint Gold," a move to implement a blockchain based digital record of bullion ownership [64]. Their idea is to back a blockchain-based digital token with one gram of gold. What makes that significant is that 1) the Royal Mint are the world's oldest gold investment organisation. Despite years of tradition, they appear to recognise the value of some very new technology, and 2) the move seems to be a form of 'gold standard' [65], a monetary system whereby a fixed quantity of precious metal backs the token of value. Historically, many currencies worked that way. However, in the early twentieth century, most countries abandoned pegging their money to gold [66]. Could the Royal Mint's move constitute a return to that?

Should the Indian government reconsider their (neutral) stance on cryptocurrencies? Might they have augmented their demonetisation scheme by looking to adopt something similar to the Forex-based kiosk, coupled with a production version of something like our currency exchange application, as shown above? That would be eminently capable of converting Indian Rupees for a cryptocurrency they instituted.

Unfortunately, The RBI's silence on cryptocurrencies means that it is unlikely they are investigating anything quite so radical as creating their version of Ether. Meanwhile, the Indian public seems more willing to accept the technology. Indeed, after demonetisation, figures suggest that there's been a rush to cryptocurrencies, with one particular Bitcoin exchange claiming that they now have over 130,000 users [67]. Although, relative to the total Indian population, that is not many; the figure is not insubstantial.

Conclusion

In this paper, we have shown an application that is capable of converting physical cash into Ether. While the technology behind the application is complex, the code employed is, actually, somewhat trivial. Moreover, there are kiosks already capable of reading foreign currencies [26] and we suggest that it is entirely plausible to imagine that such kiosks could be augmented with our application and thereby, they could offer fiat money to cryptocurrency conversion. Then we moved on to describing India's demonetisation scheme and their government's move towards digitising their financial systems and we asked whether they might have used one of those augmented kiosks to offer conversion of their banned 500 and 1,000 Rupee notes into Ether? We suggest that is unlikely because by doing so, India would give up monetary sovereignty. However, we conclude by asking whether the Indian government might have considered creating their own cryptocurrency, to which they could have offered physical cash conversion? In summary, we have shown that offering such a capacity is eminently feasible, but it appears unlikely that the Indian government has considered such a move, given the RBI's almost complete silence on all matters concerning cryptocurrencies.

About the authors

Steve Huckle is a Ph.D. research student in the School of Engineering and Informatics at the University of Sussex in Brighton, United Kingdom.
Direct comments to: s [dot] huckle [at] sussex [dot] ac [dot] uk

Martin White is Reader in Computer Science at the University of Sussex.

Rituparna Bhattacharya is a research student in Informatics at the University of Sussex.

Notes

1. N.G. Mankiw, "Principles of economics," In: . Gregory Mankiw. *Brief principles of macroeconomics*. Seventh edition. Stamford, Conn.: Cengage Learning, 2015, p. 220.

2. Ethereum, "What is Ether?" at https://www.ethereum.org/ether.

3. Ethereum, "Ethereum Project," at https://www.ethereum.org/.

4. S. Huckle, R. Bhattacharya, M. White and N. Beloff, 2016. "Internet of Things, blockchain and shared economy applications," *Procedia Computer Science*, volume 98, pp. 461–466.
doi: http://dx.doi.org/10.1016/j.procs.2016.09.074, accessed 21 February 2017.

5. Reserve Bank of India, "Press releases," at https://rbi.org.in/Scripts/BS_PressReleaseDisplay.aspx?prid=38520 (8 November 2016).

6. G. Anand and H. Kumar, 2016. "Narendra Modi bans India's largest currency bills in bid to cut corruption," *New York Times* (8 November), at https://www.nytimes.com/2016/11/09/business/india-bans-largest-currency-bills-for-now-n-bid-to-cut-corruption.html, accessed 21 February 2017.

7. M. Poppendieck and T. Poppendieck, 2010. *Lean software development: An agile toolkit, Nachdr*. Boston, Mass.: Addison-Wesley.

8. Eric Ries, 2009. "Minimum viable product: A guide" (3 August), at http://www.startuplessonslearned.com/2009/08/minimum-viable-product-guide.html, accessed 21 February 2017.

9. Solidity, "Solidity 0.4.8 — Develop documentation," at https://solidity.readthedocs.io/en/develop/, accessed 21 February 2017.

10. S. Huckle and M. White, 2016. "Socialism and the blockchain," *Future Internet*, volume 8, number 4, p. 49.
doi: http://dx.doi.org/10.3390/fi8040049, accessed 21 February 2017.

11. Ubuntu, "XenialXerus ReleaseNotes — Ubuntu Wiki," at https://wiki.ubuntu.com/XenialXerus/ReleaseNotes?_ga=1.83753119.863179621.1485520657 (7 February 2017).

12. Ethereum, "Ethereum — go-ethereum," *GitHub* https://github.com/ethereum/go-ethereum.

13. Iuri Matias, "Embark-framework," *GitHub* https://github.com/iurimatias/embark-framework.

14. React, "A JavaScript library for building user interfaces — React," at https://facebook.github.io/react/.

15. NodeJS, "NodeJs," at https://nodejs.org/en/.

16. Solidity, "Contracts 0.4.8 — Develop documentation," at http://solidity.readthedocs.io/en/develop/contracts.html.

17. CryptoCompare, "What is the 'gas' in Ethereum?" *CryptoCompare* (18 November), at https://www.cryptocompare.com/coins/guides/what-is-the-gas-in-ethereum/.

18. ISO, "ISO 4217 — Currency codes," at http://www.iso.org/iso/home/standards/currency_codes.htm.

19. H. Paul, 2015. "NIST releases SHA-3 cryptographic hash standard," *NIST* (5 August), at https://www.nist.gov/news-events/news/2015/08/nist-releases-sha-3-cryptographic-hash-standard.

20. Kraken, "Kraken | Buy, sell and margin trade Bitcoin (BTC) and Ethereum (ETH) — Buy, sell, & trade Bitcoin," at https://www.kraken.com/.

21. Gavin Wood, "Ethereum — A secure decentralised generalised transaction ledger" (EIP-150 revision), at http://gavwood.com/paper.pdf.

22. Ethereum, "Ether 0.1 documentation,&lrdquo; at http://ethdocs.org/en/latest/ether.html.

23. Ethereum, "Ethereum/web3.js," *GitHub*, at https://github.com/ethereum/web3.js.

24. Ethereum, "Ethereum/Wiki," *GitHub*, at https://github.com/ethereum/wiki.

25. Ethereum, "Account Types, Gas, and Transactions 0.1 documentation," at http://ethdocs.org/en/latest/contracts-and-transactions/account-types-gas-and-transactions.html?highlight=transaction#account-types-gas-and-transactions.

26. Fourex, "Fourex | Foreign currency exchange | Coins & notes accepted," at http://www.fourex.co.uk/.

27. Poloniex, "Poloniex — Bitcoin/cryptocurrency exchange," at https://poloniex.com/.

28. Abhinav Bhatt, 2016. "PM Modi announces notes ban in anti-corruption move, millions face cash crunch, " *NDTV.com* (9 November), at http://www.ndtv.com/india-news/pm-modi-speaks-to-nation-tonight-at-8-pm-1622948.

29. Douglas Harper, "Online etymology dictionary — Sovereign," at http://www.etymonline.com/index.php?term=sovereign.

30. Encyclopædia Britannica, 2009. "Sovereignty" (20 November), at https://www.britannica.com/topic/sovereignty.

31. Joseph Huber, "What is sovereign money?" at http://www.sovereignmoney.eu/what-is-sovereign-money/.

32. Reserve Bank of India, 2016. "Reserve Bank of India Act, 1934" (27 June), at https://rbidocs.rbi.org.in/rdocs/Publications/PDFs/RBIA1934170510.PDF.

33. F. Martin, 2013. *Money: The unauthorised biography*. London: Bodley Head.

34. Firstpost, "Demonetisation: Opposition to hold protests across India, BJP claims it's a wasted effort" (27 November), at http://www.firstpost.com/politics/demonetisation-opposition-to-hold-nationwide-protests-bjp-claims-its-a-wasted-effort-3126864.html.

35. K. Ahmed, 2016. "India raises withdrawal limit as rupee anger mounts," *BBC News* (13 November), at http://www.bbc.com/news/business-37969604, accessed 21 February 2017.

36. J. Rowlatt, 2016. "Why India wiped out 86% of its cash overnight," *BBC News* (14 November), at http://www.bbc.com/news/world-asia-india-37974423, accessed 21 February 2017.

37. V. Doshi, 2016. "Cash for queues: People paid to stand in line amid India's bank note crisis," *Guardian* (27 November), at https://www.theguardian.com/world/2016/nov/28/india-bank-lines-controversy-cash-for-queuing, accessed 21 February 2017.

38. U. Krishnan, "India to scrap two biggest bank notes at midnight to tackle corruption," *Independent* (8 November), at http://www.independent.co.uk/news/world/asia/india-currency-bank-notes-denomination-500-1000-rupees-tax-evasion-corruption-narendra-modi-a7405021.html.

39. World Bank, "The informal sector: What is it, why do we care, and how do we measure it?" at http://siteresources.worldbank.org/INTLAC/Resources/CH1.pdf.

40. B. Mazzotta, B. Chakravorti, R. Bijapurkar, R. Shukla, K. Ramesha, D. Bapat, D. Roy, N. Joseph, S. Sharan, R. Korenke and S. Durgavanshi, "The cost of cash in India," at http://fletcher.tufts.edu/~/media/Fletcher/Microsites/Cost%20of%20Cash/COC-India-lowres.pdf.

41. J. Manyika, S. Lund, M. Singer, O. White and C. Berry, 2016. "How digital finance could boost growth in emerging economies," *McKinsey Global Institute* (September), at http://www.mckinsey.com/global-themes/employment-and-growth/how-digital-finance-could-boost-growth-in-emerging-economies.

42. G. Mathew, 2014. "Check your cash, pre-2005 notes will not work after July," *Indian Express* (23 January), at http://indianexpress.com/article/india/india-others/rbi-to-withdraw-all-currency-notes-issued-before-2005-after-march/, accessed 21 February 2017.

43. *Times of India*, 2010. "Learning with the *Times*: What is Aadhaar?" (4 October), at http://timesofindia.indiatimes.com/india/Learning-with-the-Times-What-is-Aadhaar/articleshow/6680601.cms.

44. National Payments Corporation of India, "RuPay," at http://www.npci.org.in/RuPayBackground.aspx.

45. G. Gopakumar, 2011. "Visa & MasterCard gone. Rupay card, bring it on" (22 June), at http://www.moneycontrol.com/news/cnbc-tv18-comments/visamastercard-gone-rupay-card-bring-it-on_559115.html.

46. S. Nidhi, 2015. "Here's what you need to know about the Digital India initiative" (28 September), at http://www.dnaindia.com/money/report-here-s-what-you-need-to-know-about-the-digital-india-initiative-2129525.

47. PTI, 2016. "Government wants Aadhaar-enabled payment to replace debit, credit cards," *Indian Express* (2 December), at http://indianexpress.com/article/business/banking-and-finance/government-wants-aadhaar-enabled-payment-to-replace-cards-4406261/, accessed 21 February 2017.

48. Government of India. Press Information Bureau, "Pradhan Mantri Jan Dhan Yojana," at http://pib.nic.in/newsite/erelease.aspx?relid=109113.

49. pmjandhanyojana.co.in, "Pradhan Mantri Jan Dhan Yojana (PMJDY) ," *PM Jan Dhan Yojana*, http://pmjandhanyojana.co.in/.

50. *Times of India*, 2016. "Government's digital push: 11 incentives for cashless transactions" (8 December), at http://timesofindia.indiatimes.com/governments-digital-push-top-11-incentives-for-cashless-transactions/listshow/55876568.cms.

51. A. Demirguc-Kunt, L. Klapper, D. Singer and P. Van Oudheusden, 2015. "The Global Findex Database 2014 — Measuring financial inclusion around the world," *World Bank, Policy Research Working Paper*, number 7255 (April), at http://documents.worldbank.org/curated/en/187761468179367706/pdf/WPS7255.pdf.

52. Global Innovation Exchange, "Beyond cash," at https://www.globalinnovationexchange.org/beyond-cash.

53. D. Jain, 2017. "Did Jan Dhan accounts really help in money laundering post demonetisation?" (13 January), at http://www.livemint.com/Industry/PQwAM6rykFx2EDJsYzowVO/Did-Jan-Dhan-account-holders-help-in-money-laundering-postn.html.

54. *Times of India*, 2016. "400–1000% increase in digital transactions after demonetisation, says government — *Times of India*" (9 December), at http://timesofindia.indiatimes.com/business/india-business/400-1000-increase-in-digital-transactions-after-demonetisation-says-government/articleshow/55897291.cms.

55. W. Shepard, 2016. "One month in, what's The impact Of India's demonetization fiasco?" *Forbes* (12 December), at http://www.forbes.com/sites/wadeshepard/2016/12/12/one-month-in-whats-the-impact-of-indias-demonetization-fiasco/.

56. Special correspondent, 2016. "With 220mn users, India is now world's second-biggest smartphone market," *The Hindu* (3 February), at http://www.thehindu.com/news/cities/mumbai/business/with-220mn-users-india-is-now-worlds-secondbiggest-smartphone-market/article8186543.ece.

57. F. Hodiwalla and D. Aneja, 2016. "Emergence of e-wallets in India; here is how the payments industry is growing rapidly," *Financial Express* (21 November), at http://www.financialexpress.com/industry/technology/emergence-of-e-wallets-in-india-here-is-how-the-payments-industry-is-growing-rapidly/451008/, accessed 21 February 2017.

58. V. Vishwanathan, 2016. "Are crypto-currencies worth your while?" (26 September), at http://www.livemint.com/Money/CE7prSpzYoTrHRHlN2BV6H/Are-cryptocurrencies-worth-your-while.html.

59. F. Martin, 2013. *Money: The unauthorised biography*. London: Bodley Head, p. 80.

60. Global Legal Research Directorate Staff. Library of Congress, 2014. "Bitcoin survey," at http://www.loc.gov/law/help/bitcoin-survey/#denmark.

61. A. Morrison, "Blockchain and smart contract automation: Private blockchains, public, or both?" *PWC*, at http://www.pwc.com/us/en/technology-forecast/blockchain/private-public.html.

62. V. Buterin, 2015. "On public and private blockchains," *Ethereum Blog* (7 August), at https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/, accessed 21 February 2017.

63. Royal Mint, "Treasure for life," at http://www.royalmint.com/.

64. Royal Mint, "Royal Mint gold (RMG)," http://www.royalmint.com/rmg.

65. *Dictionary of American History*, 2003. "Gold standard," at http://www.encyclopedia.com/social-sciences-and-law/economics-business-and-labor/money-banking-and-investment/gold-standard.

66. W. Scroggs, 1034. "What is left of the Gold Standard?" *Foreign Affairs* (October), at https://www.foreignaffairs.com/articles/1934-10-01/what-left-gold-standard, accessed 21 February 2017.

67. S. Menon and S. Das, 2016. "Currency demonetisation effect: Why cryptocurrency is gaining currency in cashless times," *Economic Times* (8 December), at http://economictimes.indiatimes.com/news/economy/finance/demonetisation-effect-why-cryptocurrency-is-gaining-currency-in-cashless-times/articleshow/55861664.cms.

Editorial history

Received 30 January 2017; accepted 20 February 2017.

"Towards a post-cash society: An application to convert fiat money into a cryptocurrency" by Steve Huckle, Martin White and Rituparna Bhattacharya is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

Towards a post-cash society: An application to convert fiat money into a cryptocurrency
by Steve Huckle, Martin White, and Rituparna Bhattacharya.
First Monday, Volume 22, Number 3 - 6 March 2017
http://journals.uic.edu/ojs/index.php/fm/rt/printerFriendly/7410/6003
doi: http://dx.doi.org/10.5210/fm.v21i13.