ブロックチェーンと仮想落書きアプリ「Kokorobakari」

塚本　泰隆*，長野　義史**

Blockchain and App for virtual graffiti “Kokorobakari”

Yasutaka TSUKAMOTO*　and　Yoshifumi NAGANO**
(tsukamoto@profound-dt.co.jp)　(nagano@saltfish.co.jp)

*プロファウンド・デザイン・テクノロジー株式会社
**ソルトフィッシュ株式会社
*Profound Design Technology Co.,Ltd.
**Saltfish Co.,Ltd.

概要　本稿では我々が開発を計画している仮想落書きアプリ「Kokorobakari」について紹介する。Kokorobakari はイーサリアム・ブロックチェーン上で動作する DApps(Decentralized Applications:分散型アプリケーション)である。Kokorobakari は旅先での感動をその場でブロックチェーンに刻み、思い出として残すと同時に感動への感謝として地域に寄付ができるアプリである。Kokorobakari は単なる仮想落書きアプリにとどまらず、人とお金をダイナミックに動かすワールドワイドレベルの巨大なトラベラーズ・プラットフォームになる可能性を秘めている。なお、仮想落書きシステムに関してはプロファウンド・デザイン・テクノロジー社（以降、PDT 社）が日本国内で特許を取得している。

キーワード：　ブロックチェーン，イーサリアム，DApps，寄付，落書き，旅，感動,プラットフォーム

1. はじめに

近年、分散型データベースとしてブロックチェーン技術が注目を集めている。これまではブロックチェーン技術を利用したビットコインなどの仮想通貨に注目が集まっていた。しかしブロックチェーン技術は仮想通貨だけでなく様々な分野に適用できるのではないか、という期待がここ数年高まってきている。

そのような中、ブロックチェーンを適用しその効果を検証する、いわゆる POC(Proof of Concept)が多くの分野で実施されている。そこで我々もブロックチェーンを適用したら面白いであろうと思われる分野について検討してみた。なお、PDT 社は 2015 年にビットコインのマイニング用 ASIC(半導体)の設計経験があり、ブロックチェーンに関する知識は以前から持っていた。

さて、一般ユーザから見たブロックチェーンのメリットは以下の通りである。

① データが半永久的に（地球上のあちこちに）残る
② データが改ざんしにくい
③ データが消えない（消せない）
④ データのバックアップを取る必要がない
⑤ 他人への送金が容易である

上記の⑤は突っ込みどころが満載かもしれないが、とりあえず目をつむっておくこととする。

我々はまず①に着目し、ブロックチェーン適用分野を検討した（偉そうに言うほどでもないが）。しかし、そもそも半永久的にデータを残したいというケースは多数あるのだろうか？日本の公的文書でも保存すべき期間は最長でも 30 年間らしい。たった 30 年の保存のためにパブリックなブロックチェーンの資源の使うはいかがなものかと思う。もちろん今後、物理的にデータが消去可能なブロックチェーンが登場すれば話は変わるが。

色々検討した結果、①に該当するのものとして旅先での落書きがあるのではと思いついた。旅先での落書きにも様々な種類があるのだが、例えば旅先で感動し、その感動を落書きとして残す場合がある。しかしそのような落書きを建造物などに書き込むはよろしくない。とは言うものの、感動を落書きし思い出として残したくなる気持ちも理解できる。

そこで我々が思いついたのが「仮想落書き」である。実際の建造物に物理的に落書きをするのではなく、文字、画像、音声などを位置情報とともにブロックチェーン上に保存し、それを仮想的な落書きとする。そして保存した落書きは、その地を再び訪れた際に位置情報をもとにブロックチェーン上から検索され、スマホ上に表示されたり、AR 技術を用いて表示されたりする。あるいは旅行後、自宅にいながらスマホ上の地図にて訪問地を表示し、そこに落書きを表示することも可能である。また落書きデータはブロックチェーンに保存されているので半永久的に残ることとなる。

さらに、せっかくブロックチェーンに落書きを保存するのだから仮想通貨も絡めてシステムが構築できないかと検

討した。その結果、旅の感動のあとには感謝があり、感謝の先には寄付があると考えた。ただし寄付という言葉は若干上から目線な感じもするので、我々は寄付ではなく「心ばかり」と呼ぶことにした。なお仮想落書きアプリ Kokorobakari という名前の由来は、「心ばかり」である。

以降、2 章ではブロックチェーンの本質について説明する。多くの Web サイトや書籍ではブロックチェーンの本質についてほとんど説明されていない。場合によっては間違った説明をしているものも見かける。3 章では仮想落書きアプリ Kokorobakari の概要について説明する。そして最後に 4 章では Kokorobakari の可能性ついて述べる。

2. ブロックチェーンと DApps

2.1 ブロックチェーンの歴史

ブロックチェーン技術は仮想通貨の 1 つであるビットコインにおいて使用され脚光をあびることとなった。Satoshi Nakamoto と名乗る人物が 2008 年 10 月にビットコインに関する論文(1)を Cryptography メーリングリストに発表した。そして 2009 年 1 月にソースコードがリリースされ、ビットコインネットワークの運用が開始された。なお、Satoshi Nakamoto が日本人であるかどうか、あるいはそもそも誰なのかについてはいまだ謎である。

Satoshi Nakamoto の論文によると彼は以下のようなことを考えたようだ。なお以下は、彼の論文そのものの表現ではなく、我々なりの解釈を多少加えたものとなっている。

●少額の送金を考えた場合、銀行等では多額の手数料がかかるため現実的ではない。

●であれば、なんとか個人間で直接送金できないものか？

●そのためにはお金を受け取る人は少なくとも相手の預金残高（支払い能力）を確認する必要がある。

●銀行などの仲介なしに、どうやって相手の預金残高を確認する？

●そうだ、相手の通帳（他の人との取引も含めすべての取引履歴）の内容がわかればいい。

●世の中のすべての人のすべての取引履歴（送金履歴）を自分で保存しおけばいい。

●それでもいいけど、自分のコンピュータが壊れたら困るな…

●だったら、世の中のすべての人のすべての取引履歴（送金履歴）を大勢で同じ内容を保存しておけばいい。

●そうすれば自分はバックアップを取る必要もない。

●でも大勢が持つデータの内容をどうやって常に同じにするの？

問題は最後の「大勢が持つデータの内容をどうやって常に同じするのか」である。ここでは取引内容を保存したデータを「取引台帳」と呼ぶことにする。例えば A さんから B さんに 100 円の送金があった場合、送金元は A さん、送金先は B さん、送金金額は 100 円、といった情報が取引台帳に書かれる。

しかしこの「取引」は世界中でほぼ同時に発生する可能性がある。先述の A さん→B さんの送金とほぼ同時に、地球の裏側では X さん→Y さんの送金が発生するかも知れない。このとき、ある人が保存している取引台帳には、先に A さん→B さんの送金情報が書かれ、そのあとに X さん→Y さんの送金情報が書き込まれるかも知れない。一方、別の人が保存する取引台帳には、先に X さん→Y さんの送金情報が書かれ、そのあとに A さん→B さんの送金情報が書き込まれるかも知れない。これでは 2 つの取引台帳において取引の順序が異なるのでよろしくない。もちろん取引順序が異なっても問題の無いケースもあるが。

2.2 ブロックチェーンはファイル同期システム

すべての人のすべての取引に関する取引台帳を保存したデータはブロックチェーンと呼ばれている。なぜ「ブロック」なのかについては後ほど説明する。ブロックチェーンの実態は単なるファイルである。すべての人のすべての取引履歴を保存するのでファイルのサイズは非常に大きい。2019 年 4 月 17 日現在、ビットコインにおけるブロックチェーンのファイルサイズは約 213GB である。この 213GB のファイルのコピーを世界中の 1 万台以上のコンピュータがそれぞれ同じ内容で保持していることになる。ビットコインのブロックチェーンが分散型データベースとも呼ばれる理由はここにある。ただし一般に、「分散型データベース」には 2 種類存在する。1 つ目は、1 つのデータ（ファイル）を細かく分割し、分割したデータを複数のコンピュータで管理する場合である。もう 1 つは、1 つのデータ（ファイル）を複数のコンピュータにコピー（複製）して管理する場合である。ビットコインのブロックチェーンは後者の「コピー」型である。

結局、全世界で 213GB ×10,000 台 =2,130TB のディスクがビットコインのための使用されているわけである。なんとも資源の無駄使いのような気もするが、これもここでは目をつむることにする。

余談だが、Satoshi Nakamoto の論文(1)ではブロックチェーンという用語は使われていない。論文中では、chain of blocks という表現が使われている。しかし世の中では、block chain という単語になって使用されている。

ここで 1 つ疑問が生じる。ビットコインに参加するには自分のパソコンやスマホに数百 GB 単位でブロックチェー

ン全体を保存しないといけないのか？という点である。実際はそのような必要は無い。ではどうやって相手の残高を知るのかだが、ざっくり言ってしまえばブロックチェーンを保存している人に教えてもらうわけである。ただし教えてくれる人が悪者だった場合はまずいことになる。このあたりの技術はビットコインでは SPV(Simplified Payment Verification)と呼ばれている。SPV の詳細についてここでは割愛する。

さて、世界中にある 1 万台以上のコンピュータ間でどうやってリアルタイムに全く同じ内容のファイル（ブロックチェーン）を保存するのであろうか？コンピュータ間でファイルの内容を常に同じに保つことを「ファイルの同期」と呼ぶ。コンピュータの専門家からは、「なんとも幼稚であいまない定義」だとお叱りを受けそうだが、まあ、そんなに間違ってはいないと思う。

ファイルの同期をとる方法として真っ先に思いつく方法は、「早い者勝ち」である。世界中のあちこちで同時に多くの取引が発生した場合、多くの人（コンピュータ）が自分の取引をブロックチェーンに書きたいと思っているわけである。このとき、その中から誰か 1 人を早い者勝ちで決め（例えば W さん）、その人が自分の保持するブロックチェーンに自分の取引を書く。その間、その他の人（コンピュータ）は、自分の取引についてはブロックチェーンには書かず、W さんがブロックチェーンに書いた内容を自分が保持しているブロックチェーンにコピーする。こうすれば世界中に存在しているブロックチェーンの内容をすべて同じにすることが可能となる。

ここで問題となるのが、「早い者勝ち」の勝者をどのようにして決めるかである。「早い者勝ち」というからには全員に何かを競争させる必要がある。ビットコインの場合は、競争として、とある「計算問題」を採用した。この「計算問題」以下の特徴を持っている。

- ●それなりに時間がかかる。ビットコインでは約 10 分。（すぐに終わる計算問題だと、同時に計算を終える人が出てくるから）
- ●計算問題の内容が全員違う。（全員が同じ問題を解くと、同時に計算を終える人が出てくるから）
- ●「計算問題」の答えを推測することはできない。
- ●解くには時間がかかるが、解いた答えが正しいかは一瞬で確認できる。（うん、確かにあなたが一番、という検証を全員が短時間でできる必要があるから）
- ●計算の難易度を調整できる（早く計算してしまう人がでてきたら難易度あげて時間がかかるようにする必要があるから）

この「計算問題」を解く行為は「マイニング(Mining:採掘)」と呼ばれている。つまり「マイニング」の目的は、ブロックチェーンに取引履歴を書き込む唯一の人（コンピュータ）を取引ごとに決めることである。結局、ブロックチェーン・ファイルの同期をとることがマイニングの目的である。決して改ざん防止や改ざん検知が主目的ではない。ここで「取引ごとに」と書いたが、正確には少し違う。実際はある程度の数の取引がブロックという単位でまとめられ、そのブロックがブロックチェーンに書かれることになる。ブロックチェーンの「ブロック」は、この「ある程度の数がまとまった取引情報」のことである。「ブロック」単位にする理由は取引ごとに「計算問題」を解いていては取引確定に時間がかかってしまうからである。

マイニングは「鉱山などでの採掘」という意味なのだが、ファイルの同期とは全く関係のない意味である。にもかかわらずなぜマイニングなのかというと、計算問題を誰よりも早く解き取引履歴をブロックチェーンに書くとそのご褒美として新規にビットコインがもらえるからである。もう少し正確に言うと、新規にビットコインが発行され、自分宛にそれが送金されてくる。これを鉱山での金の採掘に例えてマイニングと呼んでいるのである。なぜご褒美がもらえるかというと、計算問題を解くには長い時間コンピュータを動かし続ける必要があり、その電気代が馬鹿にならないからである。ご褒美がなければ（期待できなければ）誰もマイニングはしないかも知れない。これについては議論の余地があるのだが、話題が Kokorobakari からはどんどん離れていくのでこのあたりにしておく。ちなみに「計算問題」を与えて 1 人を決めるアイデアは Satoshi Nakamoto のオリジナル・アイデアではない(2)(3)。

2.3 SHA-256 と Nonce(ナンス)

ビットコインにおけるブロックチェーンでは、先述の「計算問題」として「SHA-256＋Nonce 」方式を採用している。これについては様々な Web サイトや書籍で解説されているのでそちらを参考にして頂きたい。なお SHA-256 や Nonce の話はブロックチェーンの本質でない。本質は「誰か 1 人」を決めることである。そしてその決め方の 1 つとして SHA-256 や Nonce があり、Satoshi Nakamoto はそれを採用した、というだけの話である。だと思う。間違っている場合はご指摘頂きたい。

2.4 マイニング不要のブロックチェーン？

ビットコインではマイングに約 10 分かかるように計算の難易度が調整される。早く計算を終える人が出てきたら計算の難易度を上げるわけである。マイニングに約 10 分かかるので、取引が発生してからブロックチェーンにその取引が記録されるまでにタイムラグが生じる。このタイムラグをよしとしない取引にはビットコインは使えない。

ところでビットコインではなぜ「10 分」なのか、という

疑問が出てくる。残念ながら 10 分の根拠については見つけられなかった。1 分だと同時に手を挙げる人が出てきしまう可能性が高くなるし、60 分だと取引確定に時間がかかりすぎるので、まあ、10 分くらいかなといった感じで決められたのかも知れない。

そこで高速取引をうたっているブロックチェーンも世の中には存在する。それらの中にはマイングをしないものがある。マイニングをせずにどのようにして「早い者勝ち」で 1 人を決めているかというと、そこにはからくりが存在する。そのからくりとは、ブロックチェーンを保存するコンピュータの総数にある。

ビットコインではブロックチェーンを保存するコンピュータの総数は日々刻々と変化する。ビットコインネットワークには誰でも参加できるからである。一方、マイニングをしないブロックチェーンでは、ブロックチェーンを保存するコンピュータの総数は事前に決められている。データを保存するコンピュータの総数が事前にわかっている場合は、うまく 1 人を決める方法が存在するらしい。例えば Paxos と呼ばれるアルゴリズがあるのだが、難しすぎて我々にはよくわからない。

結局、マイニングをしないブロックチェーンではブロックチェーンを保存するコンピュータの数を変更することができない。マイニングをするブロックチェーンとマイニングをしないブロックチェーンのどちらを使えばいいのか、という話になるのだが、また Kokorobakari から話題がそれて行くのでその点については別の機会に議論できればと思う。

2.5 DApps の概要

Kokorobakari はスマートホン等で動作するアプリであるが、核となる部分はイーサリアムのブロックチェーン上で動作する DApps である。DApps は Decentralized Applications の略であり、Dapps や dApps などと表記される場合もある。イーサリアム上で動作する DApps の開発にはイーサリアム専用のプログラミング言語である Solidity を使うのが現時点では一般的である。イーサリアムはビットコインと同様に数ある仮想通貨の中のひとつである。イーサリアムを使う理由は、ビットコインのブロックチェーンには取引情報しか保存できないのに対して、イーサリアムのブロックチェーンにはアプリ（プログラム）や変数なども保存できるからである。ビットコインでもちょっとしたプログラムならブロックチェーンに保存できるのだが、複雑なプログラムは保存できない。

DApps 開発用プログラミング言語としては例えば以下のような機能があれば便利である。

・ブロックチェーンへの接続が簡素に記述できる

・ブロックチェーン上への送金が簡素に記述できる

・ブロックチェーン上の変数へのアクセスが簡素に記述できる

・ブロックチェーン上の変数に対して、アクセス権限の設定を簡素に記述できる。

・オブジェクト指向を使った記述が可能である

上記に対応した言語としてイーサリアムでは Solidity というプログラミング言語が用意されている。Solidity についてはいくつか書籍(4)(5)も出版されているのでそちらを参考にして頂きたい。

Solidity でソースコードを記述したあとは solidity コンパイラを使って実行用ファイルつまり DApps を生成する。コンパイル後のファイルはブロックチェーンに書きこまれる。そしてエンドユーザはブロックチェーン上の DApps を呼び出して実行することになる。

ブロックチェーン上の DApps を呼び出して実行すると言われても普通はピンとこないと思うので図 1 を使って説明する。

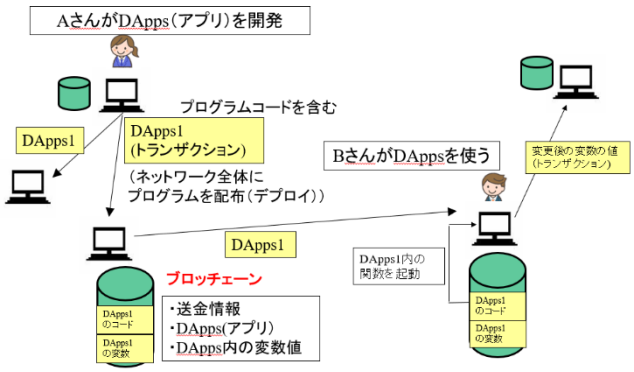

図 1　DApps の配布と起動

図 1 では A さんが DApps を開発し、B さんがその DApps を利用する様子を表している。

A さんは Solidity コンパイラを使ってコンパイルした DApps1 をブロックチェーンに書きこむ。このときブロックチェーンを保存している世界中のコンピュータに DApps1 は転送される。図 1 でいうと DApps1 は B さんにも転送され B さんが保持しているブロックチェーンに書かれる。なお 2.2 でも触れたが B さんは必ずしもブロックチェーン全体を保存するという必要はない。

B さんが DApps1 を実行する場合、B さんはブロックチェーン上にある DApps1 を起動する。起動の方法については後述する。DApps1 を起動し DApps1 を利用した結果、DApps 内の変数の値が変更された場合、変数および変更後の値がブロックチェーン上に書かれる。もちろんその変更はブロックチェーンを保存している世界中のコンピュータに伝搬する。実行結果をブロックチェーンに保存するので、イーサリアムネットワークに接続している世界中のコンピュータが DAapss1 を実行し、その結果を B さんの結果と比較する。例えば C さんがチェックのために DApps1 を実行し、もし結果が B さんと異なれば B さんが何か不正をしたとみなし、C さんはそのデータを破棄する。その結果、不正な結果はネットワーク上に伝搬することはない。

しかし同じ DApps をネットワーク上のすべてのコンピュータが実行するとはなんとも無駄な話だが、ここも目

をつむることにする。

以降、しばらくプログラミングの専門用語が出てくる。内容を理解できない人は3 章 Kokorobakari に移動して頂ければと思う。移動しても Kokorobakari を理解するには特に支障はない。

外部から書き換えるデータの例として例えば名簿を考えてみる。DApps 内で例えば名簿データを構造体の配列として記述してあるとすると、その配列データを含めてブロックチェーンに保存される。そして名簿データにデータを追加する場合は DApps を介してデータをブロックチェーン上に追加する。なお、DApps 内のすべての変数に対して新しい値をブロックチェーン上に書けるわけではない。どの変数が外部からの書き換えが可能なのかは DApps のソースコードの中で設定されている。

以上、ざっくりと DApps の配布および実行について説明したが、まだ疑問が残ると思う。GUI(グラフィカル・ユーザ・インターフェース)も DApps で記述しブロックチェーン上に保存するのか？という点である。イーサリアムの場合、現時点では無料でブロックチェーン上にデータが書き込めるわけではない。図 1 の場合、DApps1 の配布元の A さんは DApps をブロックチェーン上に配布するために仮想通貨（イーサリアム）を支払う必要がある。これをイーサリアムでは「ガス」と呼んでいる。しかも書き込むサイズが大きいほど多くのガスが必要となる。このため GUI を含む大きな DApps をブロックチェーン上に配布するのは現時点では得策ではない。ではどうするかというと通常、GUI 部分は JavaScript などを使って記述し、ブロックチェーンではなく通常のローカルディスクに置く。一方、ブロックチェーン上で管理したいデータとそのデータを管理するプログラムだけを Solidity で記述し、ブロックチェーン上に置く。その様子を図 2 に示す。

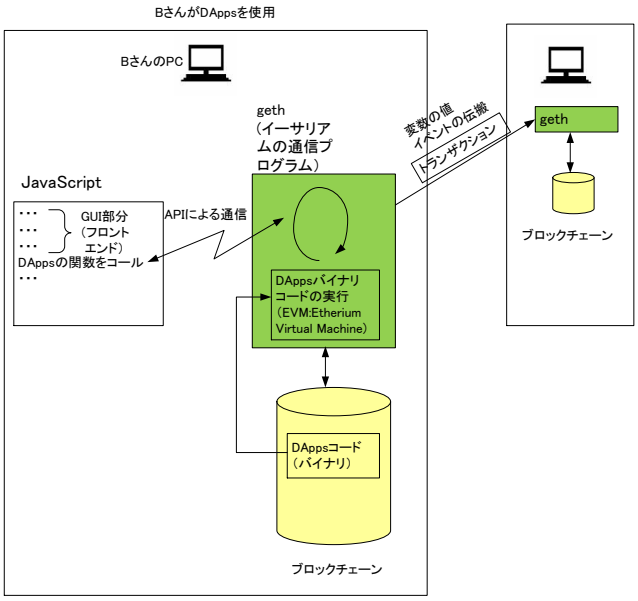

図 2　JavaScript と DApps

図 2 は B さんが DApps を使用する様子を示している。B さんはまず JavaScript で記述された GUI を起動する。そして GUI 上のメニューから DApps を起動する。B さんのコンピュータでは geth(Go Ethereum)と呼ばれるプログラムが常時動いており、隣接するコンピュータから送られてくるブロックデータを受け付ける。イーサリアムに参加する場合は geth を自分のコンピュータにインストールするだけでよい。設定によって geth はマイニングも実施する。ちなみに geth は Go Ethereum の省略形でが、Go は Go 言語を意味する。Go 言語は Google で開発されたプログラミング言語である。

DApps を起動すると、ブロックチェーンに保存されている DApps のバイナリコード(コンパイル済みプログラム)が EVM(Ethereum Virtual Machine)により実行される。これは Java の中間バイトコードが JVM(Java Virtual Machine)により実行されるのと似ている。EVM はブロックチェーンに保存されているのではなく、geth の中に存在する。

ガスの支払いが必要な人は DApps を配布する A さんだけではない。B さんは DApps を実行したとき DApps 内のデータを書き換え、そのデータをブロックチェーンに保存するのでやはりガスを支払う必要がある。

なお、実際にイーサリアムにおいて DApps の開発および運用（使用）のためには以下のようなキーワードについて理解をしておく必要がある。

- MetaMask
- Truffle
- Ganache

詳細については割愛する。

3. Kokorobakari

3.1 Kokorobakari の概要

観光地などを訪問した際、人は訪問の証として建造物などに落書きを行いたい心境になる。つまり、訪問者（観光者）はまさにその地を訪れたその時に足跡を残したいのである。落書きは人類にとって普遍的な欲望であると考えられる。しかし、遺蹟や文化財などに落書きを実際に行うことは禁止されている。

我々が開発を計画している仮想落書きアプリ Kokorobakari ではこのような欲望を満たすために、観光者が訪問場所で写真を撮り、写真に落書きを行い、そして写真を撮った位置情報（ＧＰＳデータ）及び落書き写真をセットでブロックチェーンに登録する。これを仮想落書きと呼ぶことにする。もちろん写真はなしで文字だけでもよい。落書きはブロックチェーンに書きこまれるので半永久的に残る。また自分でバックアップを取る必要もない。なお、Kokorobakari の最初のバージョンでは文字だけの保存を予定している。

観光者が地図アプリ上で該当する場所をクリック（タップ）し、その場所と写真を撮った位置情報とが一致している場合は、落書き写真がスマホなどに表示される。また、観光者が訪問場所に近づいたら、落書き写真がスマホなどに表示される。図 3 に Kokorobakari の表示イメージを示す。ここでは落書きとして文字のみを残す場合を例としている。

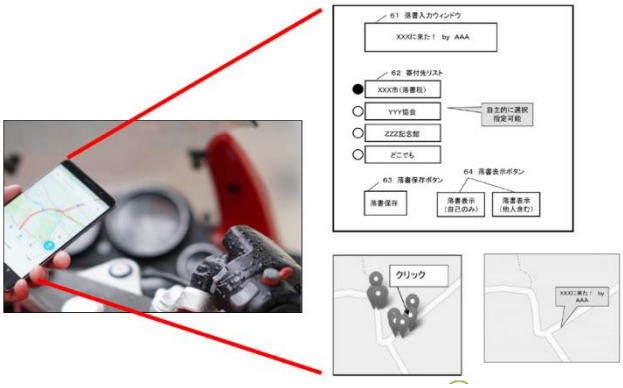

図 3　Kokorobakari の表示イメージ

また旅には感動がつきものである。感動にも様々な種類がある。景色に感動したりおもてなしに感動したり、感動の種類は人それぞれ千差万別である。そして感動の先には感謝があると考える。さらに人はその感謝の意を何等かの形で表したいと考える。感動が大きければなおさらである。Kokorobakari はその「感謝」を「心ばかり」という形で表現するお手伝いをするアプリである。Kokorobakari は「仮想落書き」をするとともに「心ばかり」の気持ちを仮想通貨の送金によって伝えることができる。

3.2 技術的課題

Kokorobakari の開発にあたっては解決すべき問題がいくつかある。例えば、エンドユーザはウォレットについて理解する必要があるという点である。仮想通貨に精通していない人にとってはウォレットの概念がわかりにくい。また、ウォレットを使う上で必要な秘密鍵およびその保管について理解する必要がある。ウォレットや秘密鍵の概念が世間一般の常識となるまでは、ウォレットや秘密鍵の概念を Kokorobakari エンドユーザからは見えないようにする工夫が必要である。

また、寄付先に対する信用性をどのようにして確保するかも課題である。自分が Kokorobakari を使って訪問先の美術館に寄付をしたと思っても寄付先が実は美術館の名を名乗るとんでもない悪者になっているかも知れない。いわゆる詐欺である。

画像などの大きなデータをブロックチェーンに保存する場合も問題となる。大きなデータをブロックチェーンに保存する場合は先述した多額の「ガス」を支払う必要がある。また、ブロックチェーン・ファイルそのものが巨大化するので、今の時点では現実的ではない。実際、Kokorobakari の最初のバージョンでは文字（テキスト）のみをブロックチェーンに保存する予定である。ただしイーサリアムにおいては Swarm と呼ばれる技術を利用すれば大きなデータも保存できる可能性がある。Swarm は大きなファイルを細かく分割し、分割したファイルを各コンピュータに分散して保存する仕組みである。2000 年代に日本でも流行った Winny とよく似ている。ちなみに Winny は今でも稼働している。

また公序良俗に反する落書き、暴力的落書き、犯罪などに対する耐性についても対策を考えておく必要がある。これは Kokorobakari に限った話ではなく、通常の SNS 等でも同じである。残念ならが現在のブロックチェーン技術では一度ブロックチェーンに書いたデータを後で削除することはできない。しかし、Kokorobakari から特定の落書きを見えなくすることは可能である。削除すべき落書きが見つかった場合、Kokorobakari からはその落書きを見えないようにする。見えないようにする方法については様々な手法が考えられるが、ここでは割愛する。削除すべき落書きをどのようにして見つけるかも問題なのだが、このあたりは AI 技術が大いに活躍しそうである。またそもそも不適切な落書きができないよう Kokorobakari が事前にチェックするということも考えられる。落書きデータとして画像や音声などを含めると不適切な落書きのチェックが難しくなるので、落書きは文字だけに限定するという考えもあるかも知れない。

3.3 Kokorobakari の利用例

図 4 に旅行会社を経由した Kokorobakari の利用例を示す。

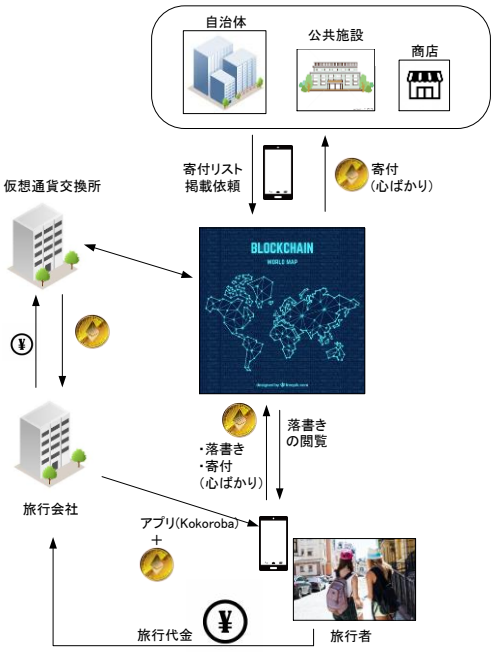

図 4　旅行会社を経由した Kokorobakari の使用例

旅行者は旅行代金を旅行会社に支払う。旅行会社はその旅行代金の一部（例えば 500 円程度）を仮想通貨に換金し、仮想落書きアプリ Kokorobakari とともに旅行者に配布する。旅行者は旅行会社から配布された仮想通貨の範囲内で寄付をする。これにより旅行者からはウォレットや秘密鍵といった面倒な事項を隠すことができる。ただし、使い切らなかった仮想通貨の扱いをどうするかという問題もある。これに対してはいくつかのアイデアがあるがここでは割愛する。

もちろんウォレットや秘密鍵を自分で管理できるユーザは自分のウォレットから寄付をすることも可能である。ただしその場合は Kokorobakari とユーザのウォレットを連動させる必要がある。

一方、寄付を希望する自治体、公共施設、商店といった人たちはスマホを使ってあらかじめ寄付リストに自分たちを登録する。登録する際は自分たちのアドレスも登録することになる。ここでいうアドレスはメールアドレスではなく、イーサリアム上でのアドレスである。

仮にある旅行会社が独占的に Kokorobakari を使用できたとしたら他の旅行会社との大きな差別化につながる可能性もある。これはあくまでも一例である。もちろん Kokorobakari をオープンソースにし、世界中の人々が自由に使えるようなことも考えられる。

4. Kokorobakari の可能性

Kokorobakari は通常の観光に限らず、山や海でも使用が可能である。例えば太平洋の真ん中を訪れ、大きな魚を釣り上げ感動し、海洋資源保全のために寄付する、といったシーンも考えられる。また、自宅にいながら Google マップ等で太平洋の真ん中を表示し、徐々に拡大していくとそこに自分の落書きや他人の落書きを見ることができる、しかもそれは半永久的に残っている。それらをみることで新たな発見や出会いが生まれそうで面白いと思う。また位置情報としては経度、緯度だけでなく高度も記録できるので落書き検索に標高や海での深さを使うこともできる。あるいはポケモン go と Kokorobakari を連動できればポケモンの出現地とそこへの寄付をからめたゲームの拡張なども可能かも知れない。

Kokorobakari のアイデアを中心として様々なアイデアが出てきそうである。我々は Kokorobakari をトラベラーズ・プラットフォームとして世の中に広めていきたいと考えている。Kokorobakari を他の人に紹介すると、こんな機能やあんな機能があれば面白いと皆さん生き生きとお話をしてくれる。もしかすると最も興味深い点は、Kokorobakari が思いやりの心で人々をつなぐ今までにない社会システムのプラットフォームにもなりえる、ということかも知れない。というのは大げさだが、まあ「話のネタ」や「酒の肴」としてはもってこいのアプリである。また、Kokorobakari の話を聞いてくださった人の中には以下のような素敵なコメントをくださる人もいる。

「

ブロックチェーンを使った Kokorobakari、なんだか早く使ってみたいですね。

『こころばかり』の寄付ならなんでも出来そうですね。

・満員電車の中で、爽やかなアナウンスの車掌さんへ

・街路樹のハナミズキがきれい。管理している市役所公園緑地課へ

・いつもキチンと挨拶してくれる保育園児。その保育園へ

こう考えると、なんだか『Kokorobakari』が世界を変えそうな感じに思えます。

」

現在 Kokorobakari は開発中であり、2019 年 12 月末にα版をリリースする予定である。

文　献

1) Bitcon:A Peer-to-Peer Electronic Cash System, https://bitcoin.org/bitcoin.pdf
2) http://www.cypherspace.org/hashcash/
3) Pricing via Processing or Combatting Junk Mail, Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology

Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings
4) Mayukh Mukhopadhyay, Ethereum Smart Contract Development(2018) Packt Publishing
5) 加嵜長門,篠原 航(2018) ブロックチェーンアプリケーション開発の教科書, Chapter 7 マイナビ出版