ブロックチェーン利用による電子書籍利便性向上のための個人間貸借管理法と実験システム試作による評価

川島悠太[1] 寺島美昭[1] 高見一正[1]

概要： 2017 年に電子書籍の売り上げが紙媒体の書籍の売り上げを上回った．本が電子化されて利便性は大幅に向上したが，それによって損なわれた機能も多い，中でも書籍の貸借に関しては「共有アカウントによる事実上の貸借」か「一部の書籍のみで期限付き」であるなど利便性が高いとは言えない．また企業の利益になりづらいという側面も持っている．本稿ではプライバシーを守りつつ企業管理ではない形で電子書籍の貸借管理が可能なシステムを設計した．管理者不在で所有者情報を管理するためにブロックチェーンを用いることによって高い改ざん耐性を持たせた．また又貸しに対応するために処理を 4 パターンに分割することで判別を可能にした．試作システムでは書籍の貸借の際に発生する手数料，処理時間から評価を行った．

Inter-individual Borrowing Management Method for Improving E-book Convenience by Using Blockchain and Evaluation by Experimental System Prototyping

YUTA KAWASHIMA[1] YOSHIAKI TERASHIMA[1] KAZUMASA TAKAMI[1]

1. はじめに

電子書籍はその普及に伴い，2017 年に漫画における紙媒体書籍の売り上げを上回った．電子書籍はその特徴として，かさばらない，1 端末のみで全ての書籍を管理できるなどの多くの利点が存在するが，利便性とともに失われたものも多い．「電子書籍を利用していたが現在は紙書籍を利用しているユーザ」へのアンケート結果[1]では「人に貸すことができないから」というユーザが 15%存在することが明らかになった．

現在，電子書籍の貸借サービスはほとんどなく，動いているサービスでも共有アカウントとして事実上の貸与を行っているユーザや貸借可能なサービスでも一部のユーザに一部の書籍しか貸与ができず，時間の制限が存在するなど利便性が低い．以前，Kindle において Lendle という書籍貸与のマッチングアプリが存在していたが企業の利益に寄与しないという理由から利用を制限している．このことから書籍の貸借運営は企業への利益とならないことが分かる．

本稿では「企業管理ではない形でプライバシーを守りつつ自由に書籍の貸借を行える貸借管理システム」を提案する．ブロックチェーン[2]を用いることで「企業管理ではない形」を実現し，管理者不在の環境でシステムを構築する．

ブロックチェーンは，電子署名，ハッシュ暗号方式，合意形成アルゴリズムなどを用いて高い改ざん耐性を実現した P2P ネットワークにおけるデータ管理方式である．履歴管理に特化していたが，ブロックチェーン上でプログラムを動作させるなどして，履歴以外のデータの管理も可能となり，現在様々な分野での応用が期待されている．本稿ではこれを電子書籍の所有者情報の管理のために用いる．

想定している利用シーンとしては，図 1 のように面識のあるユーザ二人が貸借を行い，その書籍と各ユーザの識別を行えるものをブロックチェーンに取引記録として送信する．以後，このデータをもとに貸与や返却を行う．

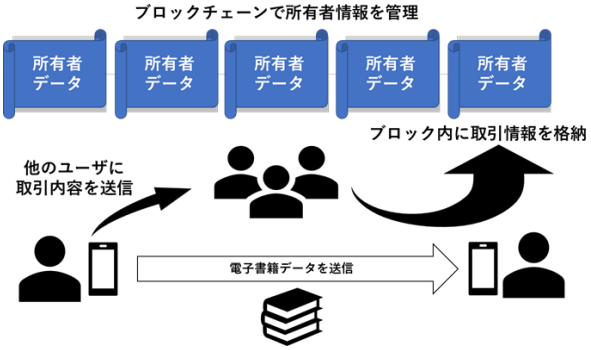

図 1 利用シーン

2. 研究課題

本研究の前提条件として，電子書籍のデータはユーザが所持しているものとし，貸借管理システムが他に存在することを考慮しない．

2.1 著作権法への配慮

そもそも書籍の貸借が著作権法などの諸法律に違反して

1 創価大学 大学院工学研究科情報システム工学専攻
Graduate School of Engineering, Soka University

いないかは重要である．本研究における著作権法の解釈としては著作権法第 26 条の 3「***著作者はその著作物(映画の著作物を除く)をその複製物(映画の著作物において複製されている著作物にあっては当該映画の著作物の複製物を除く)の貸与により公衆に提供する権利を有する***」から「公衆でなければ著作権法には抵触しない」という判断の元著作権法違反ではないと判断している．しかし，複製は認められていないため貸与後は複製が行われないようにサービスを実現する．

2.2　所有者識別のための情報選別

本研究の目的はプライバシーを守りつつ電子書籍の貸借を行うことであるため，個人の識別が可能だが，個人の特定は不可能な情報を用いる必要がある．また，貸借における必要データの選別も課題として挙げられる．ブロックチェーンでは限られたデータ容量に収める必要があるため最小のデータ量を検討する必要がある．

2.3　又貸しの検知

本研究での目的は電子書籍の貸借を行えるようにすることであるため，被貸与者が借りた書籍を別の人に貸してしまう「又貸し」と呼ばれる行為が起こらないようにするための機能を作る必要がある．

3. 提案手法

3.1　ブロック格納データ

貸借取引と又貸しを防止できるようなデータを選別してブロックチェーンにデータを格納していく．格納するデータの詳細は以下に記載する．アドレスには仮想通貨の送金する際に使用されるウォレットアドレス用いる．このアドレスは自動的に割り振られる英数字の羅列であり個人の特定が困難であること，他サービスとの連携などはあまり行われていないことからウォレットアドレスから個人情報の流出などが起こる可能性が低いと判断し採用した．

●貸与者アドレス

取引の開始時において書籍を所有しており，取引後に書籍を送信することになるユーザのアドレスを指す．

●被貸与者アドレス

取引の開始時において書籍を所有しておらず，取引後に書籍を所持することになるユーザのアドレスを指す．

●書籍ハッシュ

書籍のデータを Base64 形式の文字列に変換し，それを SHA-256 形式でハッシュに変換したものを格納する．SHA-256 形式の場合は衝突を考慮しなくてよく，ハッシュ暗号化方式はブロックチェーン技術に使用されていることからも扱いやすいと判断した．また，SHA-256 形式なら書籍がどのようなデータ量のものでも一定サイズのデータ量に変換してくれるため，疑似的な書籍 ID という形で利用する．

3.2　又貸し防止

貸与した書籍が返却されるまでの処理を「登録，貸与，返却，又貸し」4 つの分類分けを行うことで又貸しを防止する．図 2 における貸与コントラクトとは，貸与時に利用するスマートコントラクトのことであり，スマートコントラクトとはブロックチェーン上に格納されているプログラムのことを指す(以下コントラクト)．図 2 で使用している変数を表 1 に記述した．以後の各パターンの説明でも用いる．

表 1　フローチャート変数一覧

内容	変数名
今回貸与者(ユーザ A)	ID.1
今回被貸与者(ユーザ B)	ID.2
過去貸与者	PastID.1
過去被貸与者	PastID.2
今回の書籍のハッシュ	Hash
過去の書籍ハッシュ	PastHash

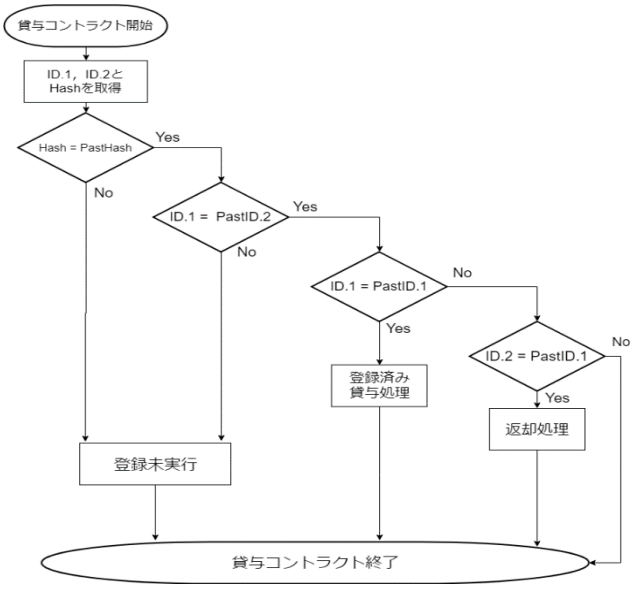

図 2　又貸し防止フローチャート

・登録(図 3)

この処理に必要になるのは ID.1 と Hash である．登録は書籍の貸与を行う前に書籍の所有権をブロックチェーン上に保存するために行う操作である．この操作でブロック内にとある書籍のハッシュ値(Hash)において貸与者アドレスと被貸与者アドレスが一致するデータを格納する．

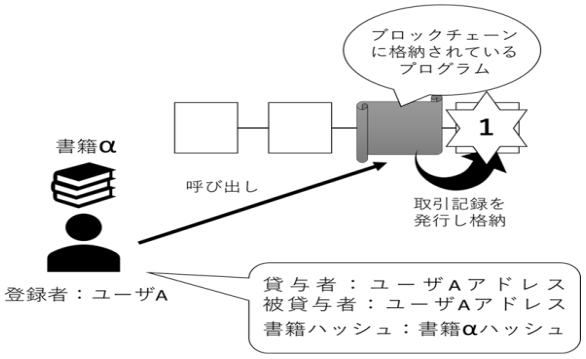

図 3　登録処理概要

登録時の処理を以下に示す.

STEP1：書籍ファイルを選択する

STEP2：コントラクト内の登録コントラクトを呼び出し，ID.1 と Hash を渡す

STEP3：コントラクトが「貸与者：ID.1，被貸与者：ID.1，書籍：Hash」として取引記録を発行する.

・貸与(図 4)

貸与の際に使用するのは，ID.1，ID.2，Hash の 3 つである．貸与の条件として，Hash と PastHash が一致するブロックにおいて，ID.1 と PastID. 2 が同一かつ ID.1 が PastID.2 とも同一であることを条件とする．これは登録の際，PastHash おいて PastID.1 と PastID.2 が一致するデータが格納されているのでそれを確認している．

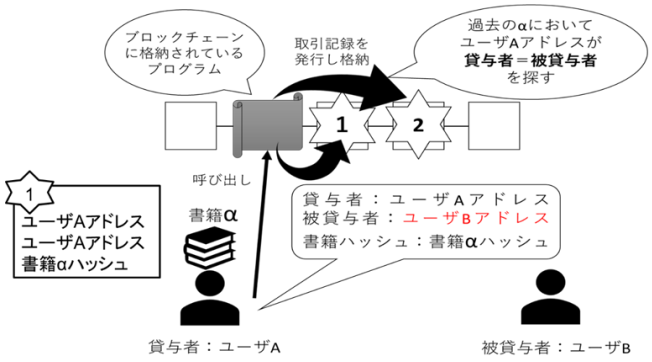

図 4　貸与処理概要

貸与時の処理を以下に示す.

STEP1：宛先アドレス(ID.2)を明記し，書籍ファイルを選択

STEP2：貸与者はブラウザからコントラクトの貸与コントラクトを呼び出し，ID.1，ID.2，Hash を渡す

STEP3：コントラクトは値を受け取り受け取ったデータを過去のデータと判別する

STEP4 : Hash と同一の PastHash を探索する

4.1 : ID.1 が貸与者及び被貸与者の過去データを探索

4.1.1：存在していれば STEP5 へ

4.2.2：存在していなければエラーを返し終了

STEP5 : ID.1，ID.2，Hash をブロックに格納しコントラクトの実行を終了する

STEP6：ブロックの格納を確認し，書籍データを送信する

STEP7：書籍の受け取り完了メッセージを確認し，貸与者の書籍データを削除する.

・返却(図 5)

返却の際にも貸与時と同様に ID.1，ID,2，Hash の 3 つのデータを用いて行う．返却は Hash が PastHash と一致しているブロックにおいて，ID.1 が PastID.2 と一致し，ID,2 が PastID.1 と一致するブロックデータが存在していることを条件としている．この条件を満たしているブロックが存在する場合のみ貸与処理は返却処理として扱われる．

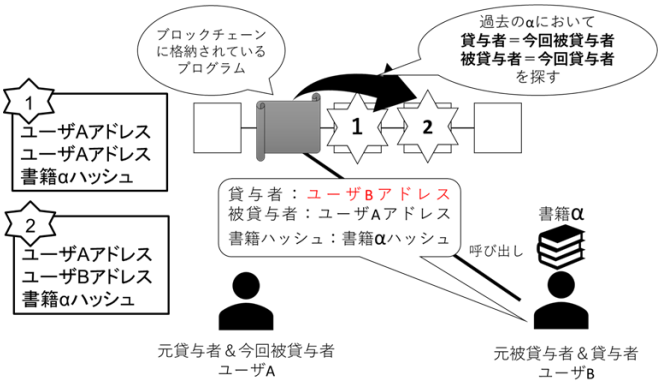

図 5　返却処理概要

返却時の処理を以下に示す.

STEP1：宛先アドレス(ID.2)を明記し，書籍ファイルを選択

STEP2：貸与者はブラウザからコントラクト内の貸与コントラクトを呼び出し，ID.1，ID.2，Hash を渡す

STEP3：コントラクトは値を受け取ったデータと過去のデータを参照し判別を開始する

STEP4 : Hash と同一の PastHash を探索する

4.1：一致すれば STEP5 へ

4.2：不一致の場合は他の過去データを探索し，一致データが存在しない場合はエラーを返す.

STEP5 : ID.1 が PastID.2 と同一である過去データを探索

5.1：一致していれば STEP6 へ

5.2：不一致かつ上記の貸与にも該当しない場合はエラーを返す

STEP6 : ID.2 が PastID.1 と同一である過去データを探索

6.1：一致していれば STEP7 へ

6.2：不一致の場合はエラーを返す

STEP7：返却の処理として貸与時にブロックに格納した「ID.1，ID.2，Hash」という配列番号に NULL を格納し，今後の処理を邪魔しないようにする.

STEP8：書籍データを送信し，貸与者の書籍データを削除

・又貸し

上記 3 つのパターンに当てはまらない状態全てを又貸しとして処理する.

3.3　ファイルの削除機能

2.1 節で明記した著作権法への配慮から，不特定多数への貸与が行えるようなシステムは違法であると判断しているため特定少数のみに貸与が可能なシステムを構築する．特定少数とは具体的には，最低限顔は知っている程度の間でのみで利用可能なシステムとし，著作権法は主に貸与より複製の際に高頻度で問題視されることから取引後に書籍データが複製されていてはいけない．貸与者と被貸与者両方に書籍データが残るようなことにならないよう送信完了後にファイルを削除する機能を実装する.

4. 試作システムと実験環境

提案の有用性を評価するために試作システムを構築する．P2P モデルでシステム構築を行うため，それぞれのノードはクライアント機能とサーバ機能を有しており，クライアント PC がブラウザを介することでコントラクトにアクセスし，提案手法の処理を実行可能にする．図 6 に試作システムの概要を示す．電子書籍の閲覧が最も多いのはスマートフォンだが，良好な通信環境でのデータを取得するため有線を用いたローカルネットワークを用いて PC による実験を行う．試作システムの構成表は表 2 に示す．コントラクトとの連携には Web3.js というライブラリ機能を利用しなくてはならないことから，同一言語で実装でき，双方向の Socket 通信が容易に実装できることからサーバ機能とクライアント機能は Node.js で実装した．ブロックチェーンの開発基盤は Ethereum[3] を用いてテストネットには Ethereum と同一環境で実験を行える Ropsten ネットワーク [4] を利用した．ウォレットの秘密鍵の管理機能には GoogleChrome の拡張機能である Metamask を用いた．それに対応するブラウザを用いる必要があったため使用ブラウザは GoogleChrome と Firefox とした．

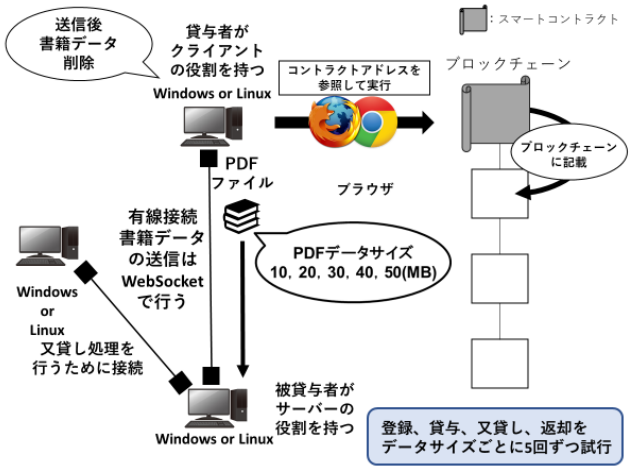

図 6　試作システム及び実験環境評価

表 2　試作システム構成図

項目	内容
使用 OS	Windows10 pro Ubuntu18.04(VirtualBox) Ubuntu16.04(32bit)
サーバクライアント	Node.js
プログラム記述言語	Web ページ：HTML，Javascript スマートコントラクト：Solidity
ブロックチェーンネットワーク	Ropsten ネットワーク
ウォレット機能	MetaMask
ブラウザ	Google Chrome，Firefox
ネットワーク環境	有線ローカルネットワーク

5. 実験手順と評価条件

評価実験の流れを以下に記述する．実験条件は表 3 とする．

・登録

STEP1：書籍を所有している PC がブラウザを立ち上げる．

STEP2：ブラウザ上から書籍を選択し登録コントラクトを呼び出す．

STEP3：コントラクト実行のための手数料が表示され，許可を出しコントラクトが動作する．

STEP4：コントラクトが書籍をハッシュに変換し，取引記録を発行する．

STEP5：発行された取引記録に対して発生した手数料を確認し，取引記録を送信する．

STEP6：ブロックへの格納を確認する．

・貸与

STEP1：書籍を所有している PC がサーバ機能，書籍受け取り側がクライアント機能を立ち上げる．

STEP2：クライアント PC がサーバ PC に接続する．

STEP3：クライアント PC のウォレットアドレスを入力し，書籍を選択する．

STEP4：貸与コントラクトを呼び出す．

STEP5：コントラクト実行のための手数料が表示され，許可を出しコントラクトが動作する．

STEP6：コントラクトは登録情報が格納されているかどうかを確認し，取引記録を発行する．

STEP7：発行された取引記録に対して発生した手数料を確認し，取引記録を送信する．

STEP8：取引記録がブロックに格納されたことを確認する．

STEP9：書籍を送信する．

STEP10：サーバ PC の書籍データを削除する．

・又貸し

STEP1：貸与時に書籍データを受け取った PC がサーバ機能，先ほどの取引では関与していなかった PC でクライアント機能を立ち上げる．

STEP2：クライアント PC がサーバ PC に接続する．

STEP3：クライアント PC のウォレットアドレスを入力し，書籍を選択する．

STEP4：貸与コントラクトを呼び出す．

STEP5：コントラクト実行のための手数料が表示され，許可を出しコントラクトが動作する．

STEP6：コントラクトが登録情報を確認するが，存在しない為エラーを取引記録として発行する．

STEP7：取引が終了し，ブロックへの情報の格納や書籍の移動は行われない．

・返却

STEP1：書籍を所有している PC がサーバ機能，書籍受け取り側がクライアント機能を立ち上げる．

STEP2：クライアント PC がサーバ PC に接続する．

STEP3：クライアント PC のウォレットアドレスを入力し，書籍を選択する．
STEP4：貸与コントラクトを呼び出す．
STEP5：コントラクトは貸与情報が格納されているかどうかを確認し，返却処理であると判断して取引記録を発行する．
STEP6：発行された取引記録に対して発生した手数料を確認し，取引記録を送信する．
STEP7：取引記録がブロックに格納されたことを確認する．
STEP8：書籍を送信する．
STEP9：サーバ PC の書籍データを削除する．

以上の動作をデータサイズごとに 5 回ずつ行う．評価項目については，まず Ethereum を基盤として利用したサービスを検討する中で，必ずユーザ負担となって発生する手数料の存在がユーザの利用障壁になる可能性の有無を調べるために手数料評価を行った．また，そのほかに利用障壁となる可能性がある処理時間を評価項目として設定した．

表 3　実験条件

項目	値
データサイズ(MB)	10,20,30,40,50
試行回数	各データサイズ 5 回
実験パターン	登録，貸与，返却，又貸し
評価項目	システム動作 手数料 処理時間

6. 評価結果

5 章の実験手順で評価実験を行ったところ，各データサイズにおいて 5 回の試行で 5 回とも書籍の著しい劣化や送信エラーなどがなく貸与や返却を行うことができた．また，貸与，返却後は書籍の削除も問題なく行われた．これらの結果から本研究の目的である「企業管理ではない形でプライバシーを守りつつ自由に書籍の貸借を行える管理システムの実現」が行えたということが分かる．その上でサービスとしての評価を行う．

・手数料

実験での手数料を図 7 示す．図 7 はデータサイズにおける実際に残高から引かれた額を記録したものである．5 回行った各パターンの実験結果の平均値をグラフとして示している．Ether とは Ethereum 上で使用されている通貨の単位である．実験時は 1Ether＝10000 円程度であったため，図 7 における縦軸の最大値はおよそ 2.5 円程度である．図 7 を見ると手数料としてかかった金額は一番高い平均値となっている 10MB の貸与時でも 3 円に満たない．また，データサイズごとに見てもデータサイズによって手数料が大きく上下する様子は見られない．図 7 において 10MB の登録時と貸与時の手数料が大きく上昇している理由としては，Ethereum の手数料決定の仕方が影響している．Ethereum における手数料の決定方式はブロックへ格納される取引の数に依存している．10MB の登録と貸与の試行回数ごとの発生手数料の発生額は表 4，表 5 のようになっている．同一の試行回数において登録と貸与の請求 Ether が他の試行回数の 3 倍近い額となっている．請求 Ether に対して大体 66%が実費 Ether となっているため請求額が上昇すると実費額も相対的に上がっていく．Ropsten ネットワークの手数料の上限は決まっているためそれに格納する取引記録を発行するタイミングによって，取引の数にまれではあるが変動があるため，個人が負担する手数料の金額が変化する．Ethereum や Ropsten ネットワークにおけるブロックへの格納サイズは格納データ上限によって決定されているため，このような変動が起こりうる．しかし，それを踏まえても各取引にかかる費用は 5 円以下であるため，「2012 年から 2017 年の日本人 1 人当たりのスマートフォンアプリへの課金額が 214 ドルである」[5]という調査結果を考慮するとサービスを利用する際にユーザの利用障壁になるような額ではないと考えられる．

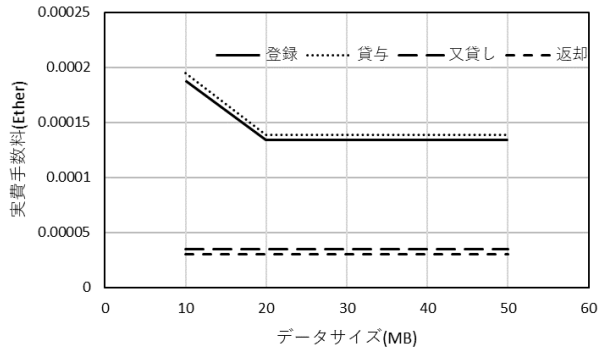

図 7　パターンごと手数料比較

表 4　10MB 登録手数料

試行回数	請求 Ether	実費 Ether	請求円	実費円
1 回目	0.000201	0.000134	2	1
2 回目	0.000201	0.000134	2	1
3 回目	0.000201	0.000134	2	1
4 回目	0.000603	0.000402	6	4
5 回目	0.000201	0.000134	2	1

表 5　10MB 貸与手数料

試行回数	請求 Ether	実費 Ether	請求円	実費円
1 回目	0.000209	0.000139	2	1
2 回目	0.000209	0.000139	2	1
3 回目	0.000209	0.000139	2	1
4 回目	0.000627	0.000418	6	4
5 回目	0.000209	0.000139	2	1

・処理時間

各パターンの処理時間を図 8，図 9，図 10 で示す．全てのグラフに言えることは，データサイズが大きくなるにつれて取引の所要時間が長くなってしまっていることである．この原因は書籍データをハッシュ文字列に変換する時間が書籍サイズの上昇によって長くなっている為である．今回の書籍データの変換方法は PDF ファイルである書籍データを一度 Base64 形式の文字列に変換し，そこで出来上がった文字列を SHA-256 方式のハッシュ関数にかけて書籍のハッシュ値を導出している．そのためデータサイズが大きければ大きいほどその変換に時間がかかってしまっている．また，登録と貸与の変換時間に比べて返却の変換時間が 2 倍程度かかってしまっているのは CPU の問題であると考えられる．返却の際はクライアント側が VirtualBox 上で起動させた Ubuntu18.04 を使用しているが，シングルコアのプロセッサでメモリは 4 GB で動作させている．そのため登録時や貸与時に利用している PC よりもスペックで劣っている為，それらの変換時間に比べて返却の際に変換の時間が長くなってしまっている．結果，取引時間がユーザの CPU の性能に依存するものとなってしまっている．

各パターンのデータサイズ別所要時間を全て合計して各パターンでの平均所要時間を表 6 に算出した．ブロック格納時間は 45 秒程度の時間がかかってしまっている．Ethereum における 1 ブロック格納は約 15 秒であることから確定までに 3 ブロック分は待つ必要があることが分かる．しかし，本稿のような所有権の譲渡が行われる取引の場合は取引記録が格納されるものという予測で動いてしまうと格納されなかった時に収集がつかなくなってしまうため決済の完了を待ってからのデータの譲渡が必要である．本稿では許容手数料設定が最低値であったためこれだけの時間を要しているがこれを引き上げることで時間を短縮することは可能である．

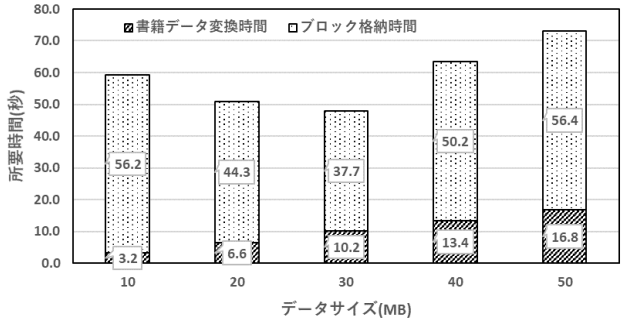

図 8　登録処理時間

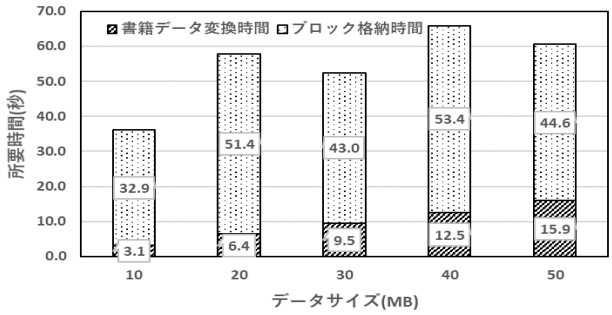

図 9　貸与処理時間

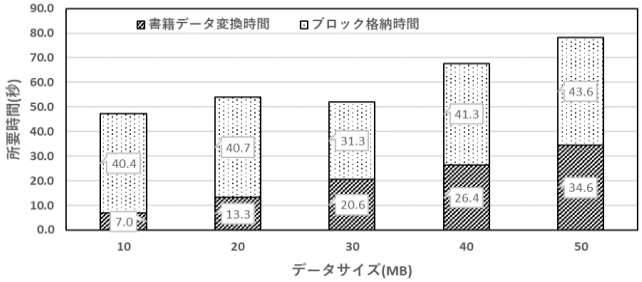

図 10　返却処理時間

表 6　各パターン所要時間

工程	登録(秒)	貸与(秒)	返却(秒)	平均(秒)
変換時間	10.0	9.5	20.4	13.3
格納時間	48.9	45.1	39.5	44.5
合計時間	59.0	54.5	59.9	54.4

7. まとめ

本稿では「管理者となる企業などの中央管理サーバを用いずに，プライバシーを守りつつ自由に電子書籍の個人間の貸借を行うことができる」サービスシステムの提案から試作システムを作成し評価を行った．結果ブロックチェーンを用いて電子書籍の貸借管理を行うことが可能であることが確認できた．また，手数料を考えても登録，貸与，返却までで 10 円に満たないという良い結果を得ることができた．しかし，データがブロックに格納されるまでに 30 秒～80 秒までかかってしまっていることがユーザの大きな利用障壁になってしまうと考えられるため，全体の処理時間の短縮は大きな課題となっている．その他にも，現行サービスの連携や複数アカウントへの対応，借りたまま返却を行わないユーザへの対策などを講じる必要がある．

参考文献

[1] マクロミル「電子書籍に関する調査」2016 年
[2] Satoshi Nakamoto「Bitcoin:A Peer-to-Peer Electronic Cash System」2009 年
[3] Ethereum(参照 2019-4-25)　https://www.ethereum.org/
[4] Ropsten(参照 2019-04-25)　https://blockscout.com/eth/ropsten/
[5] SensorTower(参照 2019-4-23)「Japan Leads Per Capita App Store Spending at More Than $200 Per Person Since 2012　https://sensortower.com/blog/per-capita-app-store-spending