CasualChain
技術概要

2024/08/03 亀井亮児

What is CasualChain?

A blockchain core system that doesn't have any incentive features inside
- Man shall not live by crypto currency alone

インセンティブ機能を内包していない、ブロックチェーンコアシステム
- 人は暗号資産のみに生きるにあらず

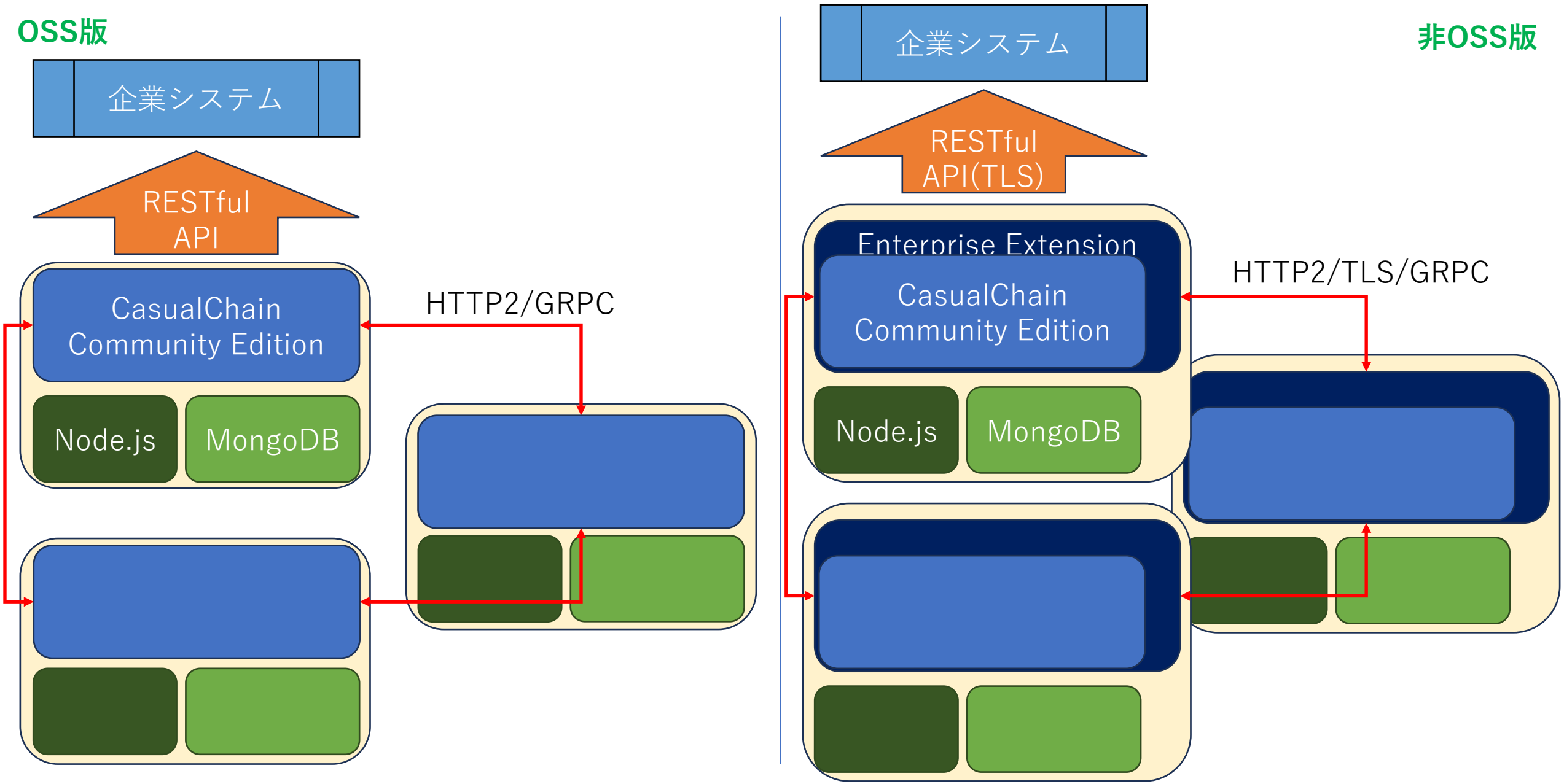
[READMEより]

一般的な企業システム(※)に**カジュアルに**組み込まれて使われることを想定した、プライベートブロックチェーンエンジンです
①暗号資産に依存せずにブロックチェーンの恩恵を受けることができます
②REST APIを用いた簡単アクセス。特定のプログラミング言語に依存しません
③ブロックチェーンの専門用語に依存せず、一般的なDB用語を用いて理解できます
④ライセンス：主要部分はMIT、拡張部分は非営利無料ライセンス(PUEL) ※後述
⑤インセンティブ機能がないため、不特定多数では利用できません(各ノードでの電子署名機能のみ)
以下のコンポーネントがインストールされたLinuxシステム上で動作します
- Node.js (18もしくは20)
- OpenSSL
- Git
- MongoDB (4.4以上)

OSS版(Community Edition、CE)と非OSS版(Enterprise Edition、EE)が存在します

※企業システム：toB/toC限らず、特定の社会的役割を与えられたITシステムのことを指します

システム構成


OSS版
企業システム
RESTful
API
CasualChain
Community Edition
Node.js
MongoDB
HTTP2/GRPC
非OSS版
企業システム
RESTful
API(TLS)
Enterprise Extension
CasualChain
Community Edition
Node.js
MongoDB
HTTP2/TLS/GRPC

OSS版(Community Edition)と非OSS版(Enterprise Edition)の主な機能と比較

主な機能	OSS版 (CE)	非OSS版 (EE)
トランザクションの追記型記録	○	○
大きなトランザクションへの対応(1トランザクション最大15MBまで)	○	○
ブロック化待ちトランザクションを一時プールする機構とプール中トランザクションの透過的な取り扱い	○	○
オリジナルブロック化アルゴリズム	○	○
ノードごとのアクセス権の設定	○	○
不正なノードとの通信のブロック	○	○
複数トランザクションの一括ブロック化	○	○
楕円曲線暗号を採用しデータ量削減と高速化	○	○
RESTful API	○	○
HTTP2ノード間通信	○	○
トランザクション間の履歴チェーンを簡単に構築可能	○	○
改ざん・枝分かれ・ノード間データ不整合の検知	○	○
改ざん・枝分かれ・ノード間データ不整合の修復	○	○
自動的なブロック化等、内部タスクのスケジューリング設定	○	○
ブロック化待ちのトランザクションのディスクへの書き出しによる保全が可能		○
オンメモリトランザクションI/Oが可能		○
任意のタイミングでのブロック化等、より高度なAPIが利用可能		○
RESTful APIおよびノード間通信のTLS利用可能		○
マルチテナント (※一本のチェーンを複数のテナントで安全に共有し、チェーン強度を高める)		○

REST風 API

ユーザー系	CE	説明
/get/byjson	○	KVペアでヒットするデータをJSONで取得
/get/byoid/:oid(¥¥w{24})	○	oidを指定して特定データをJSONで取得
/get/alltxs	○	全てのトランザクションを取得(ブロック化待ちのトランザクションを含む)
/get/blocked	○	ブロック化された全てのトランザクションを取得
/get/pooling	○	プール中の全てのトランザクションを取得
/get/poolingdelivered	×	プール中の全てのトランザクションのうち、既に他ノードへ転送済みのもののみを取得
/get/lastblock	×	チェーンの最後のブロックを取得
/get/totalnumber	×	総ブロック数を取得
/get/history/:oid(¥¥w{24})	○	指定したトランザクションより前の一連の繋がりあるトランザクションを取得
/post/byjson	○	JSON形式でデータを1件登録

非OSS版(EE)では、GET系は自分のテナントのデータのみを取得。またPOST系は自分のテナントIDを設定する

管理系	CE	説明
/sys/initbc	○	ブロックチェーンの初期化(再初期化は原則できない)
/sys/opentenant	×	新規テナント区画の確保
/sys/closetenant	×	既存テナント区画の閉鎖
/sys/deliverpooling	△	プール中のトランザクションを他ノードに転送
/sys/blocking	△	同期済みのトランザクションをブロック化
/sys/syncblocked	○	ブロック化トランザクションの改ざん・枝分かれ・ノード間データ不整合の修復
/sys/syncpooling	○	プール中のトランザクションのノード間データ不整合の修復
/sys/synccache	×	オンメモリトランザクション時にキャッシュとDB間を強制的に同期

OSS版(CE)での扱い
○：利用可能
△：自動的に実行のみ
×：利用できない

MongoDBのライセンス縛りについて

❗ MongoDBはオープンソースではないプロダクトで用い、かつサービスとして提供する場合は、商用ライセンス(有償版もしくはクラウド版)を使用する必要があります

ケース	OSS版(CE) をそのまま使用もしくはカスタマイズしてOSSとして公開	非OSS版(EE)を使用、もしくは非OSSカスタマイズして使用
システム納入先が社内で使用	MongoDB無償版利用可能	MongoDB無償版利用可能
自前サービスを構築し第三者に提供	MongoDB無償版利用可能	MongoDB商用ライセンス必要
システム納入先がサービスを提供	MongoDB無償版利用可能	MongoDB商用ライセンス必要

したがって、利用者からみた場合、CasualChain CE自体は緩いMITライセンスであるものの、MongoDBにお金を払わないで済ませるにはCasualChainのカスタマイズ部分のOSS公開が必要であり、全体としてはGPLに近いライセンス取り扱いになる

今後の検討課題・アイデア

次期バージョン以降

- 再設計：他のプロジェクトに組み込めるよう、コアのライブラリ化
- 設定の動的変更に正式対応
- トランザクション履歴の併合に対応
- HTTP2対応高速API
- MongoDB以外のデータストアへの対応(費用の節約・EEのみ)
- スケールアウトに対応など、クラウドネイティブ機能の拡充
- テナントごとの認証設定
- PoAの変形版を開発しパブリックチェーンに対応開始　※インセンティブベースではない

IPFSとの連携(別サービス)

- より大きなファイルの扱いに対応
- 実行ファイルの配布に対応(いわゆるスマートコントラクト相当)