Reference Stateの活用方法
(トレーサビリティ領域)

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

自己紹介

井本 翔太（いもと しょうた）

- SBI R3 Japan プロダクトサービス部

＜略歴＞

愛知県出身: 20歳(2/27)

大学生(都内)　: 2020~

SBI R3 Japan　: 2021(12月 join)~

今までCorda trainingの改修や今回の調査など行ってきました!

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

SBIr3 Japan

はじめに

はじめに

- ReferenceStateに関するサンプルを作成しました
- ReferenceStateの細かい挙動を確認することが目的でした。
- 技術的な内容についてご相談します。

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

Reference Stateとは？

- Reference Stateとは参照情報を表すStateの事です

- Corda公式ドキュメント
  - ✓ https://docs.r3.com/en/tools/cdl/ledger-evolution-view/reference-states.html

- Cordaトレーニングドキュメント
  - ✓ https://training.corda.net/corda-advanced-concepts/reference-states/#:~:text=Reference%20states%20are%20part%20of,usual%2C%20verification%20outcomes%20are%20deterministic.

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

ビジネスの紹介（SHIMENAWA@SBIトレーサビリティ）

段階	POC　Pilot　Production
ソリューション名	SHIMENAWA
ユースケース	トレーサビリティ、真贋証明
開発会社	SBIトレーサビリティ株式会社 (Digglue, ITForce他)
ユーザー	米：北海道の大手コメ卸他 酒：加藤吉平商店（梵BORN）他 農産物全般：農水省スマートフードチェーンコンソーシアム
国	日本発グローバル
背景、課題	• 履歴提供に対する消費者ニーズの高まり • 温暖化抑止に対する機運の高まり • 日本の飲食品輸出拡大戦略 • 不正品抑止
特徴、利点	1. ｻﾌﾟﾗｲﾁｪｰﾝの透明性実現 2. 真贋証明による新たな信用力確立 3. 模倣品不正流通の抑止 4. 正規品不正転売の抑止

消費者向けアプリイメージ

TVニュースで取り上げ

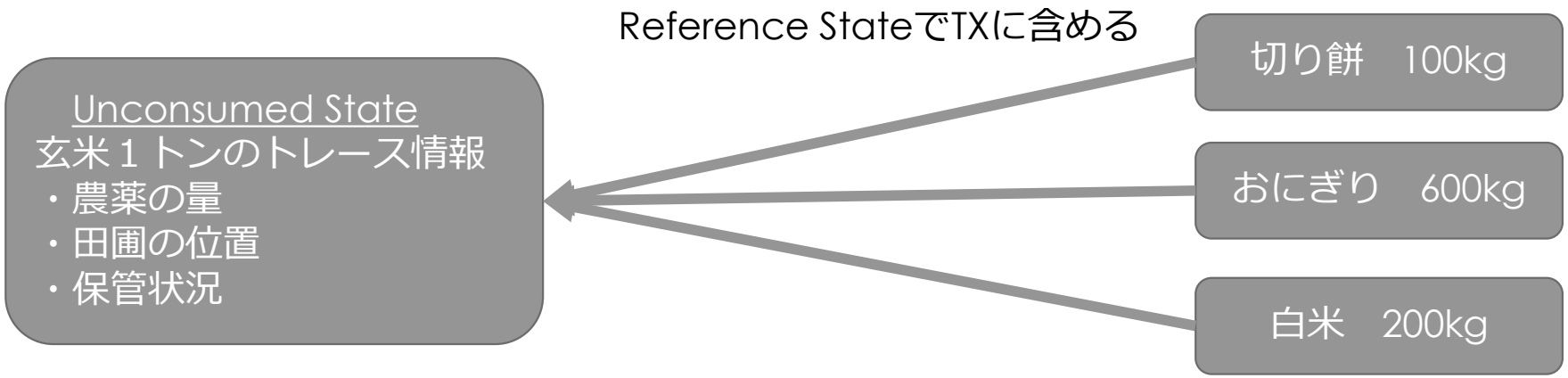
TVニュース動画へのリンク　https://www.fnn.jp/articles/-/351751

Next Step	• 2022年6月 商用化（米） • 2022年度2Q 商用化（酒、真贋判定） • toC向けアプリUXの改善 • 汎用API / 汎用Cordapp の構築

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

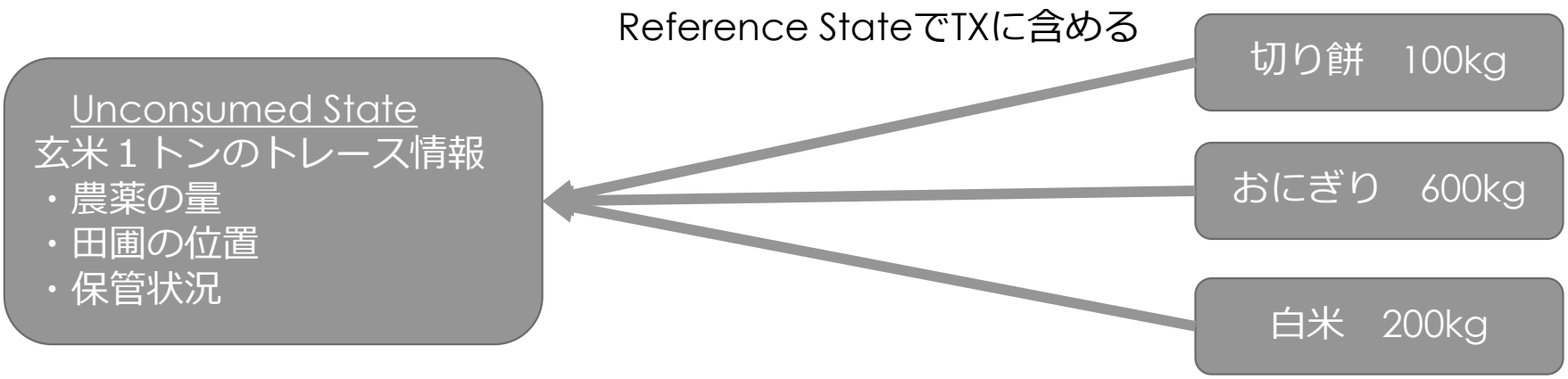
調査の背景（トレーサビリティにおけるReference Stateの事例）



Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

調査課題：技術的課題

Reference StateはInput Stateと同じように動作するのか？
（課題１）TXに含めたRef. Stateをスマートコントラクトで検証可能？
（課題２）Ref. Stateの過去履歴（Transaction Backchain）は共有されるのか

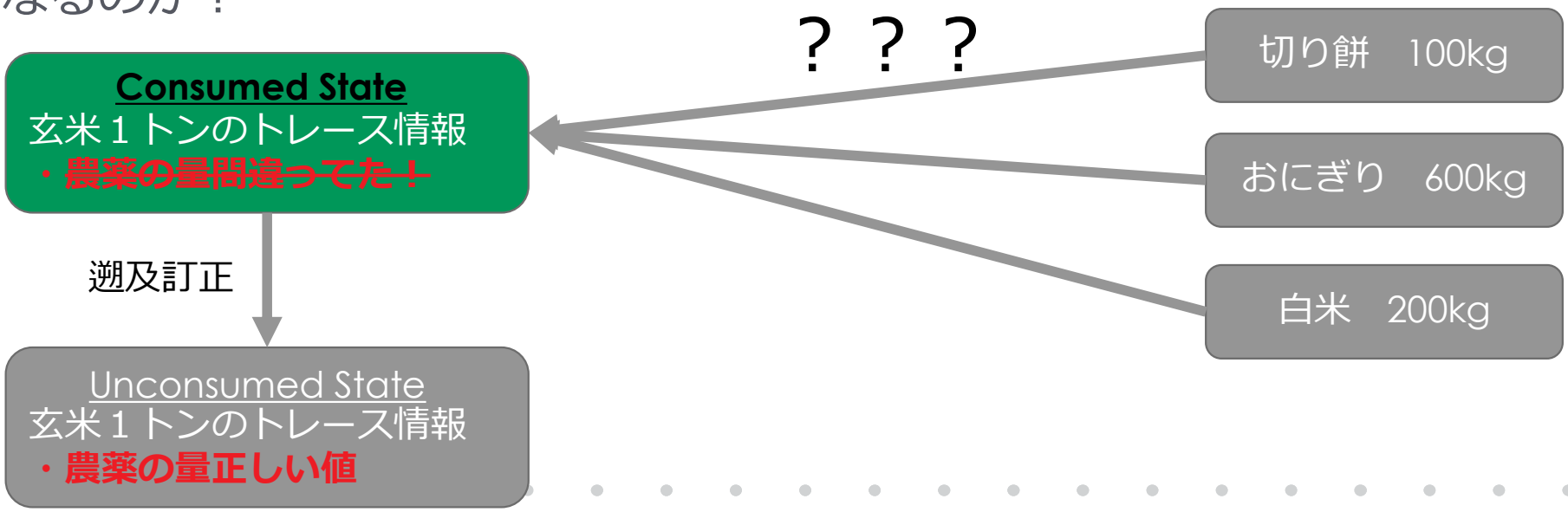


Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

調査課題：ビジネス的な課題

おにぎりを作った後で、玄米情報を書き換える（遡及訂正）場合どうするのか？

（課題３）
　ConsumedState（＝書き換え前の情報）をReference Stateとして含めた場合どうなるのか？



Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

SBIr3 Japan

調査内容と結論

調査内容

- 1. Ref.Stateをトランザクションの中に含めるか？
  また、通常のコントラクトの中で検証できるか？

- 2. あるノードが発行したRef.Stateのバックチェーンを
  別のノードで確認できるのか？

- 3. 一つ前のRef.Stateをトランザクションに含められるか?

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

結論

- 結論から申しますと、要望1,2は満たすことができましたが要望 3 はNotaryがエラーを出しました。
  - ✓エラーが出る＝期待通りの動作になります
    - ➢ 理由は後述します

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

SBIr3 Japan

サンプルプログラムについて

サンプルプログラムの説明

- Corda trainingのIOUコード(issueのみ)に関して住所録(AddressState)をRef.Stateとして追加しました

- なぜissueのみか？
  - ✓TransferやSettleでRef.Stateの紐づけを行うとIOUの発行者以外の住所録が必要になり、実装も複雑になります
    - ➢ 与えられた時間を考慮した結果、このような形を取りました

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

作成したファイル

- AddressState.java
  - ✓ Ref.Stateに相当するStateになります
- AddressContract.java
  - ✓ AddressStateを発行するPublishコマンドとAddressStateを更新するMoveコマンドを定義し、それらに関する制約を加えました
- PublishFlow.java
  - ✓ AddressStateの発行(Publish)に関するFlowを作成しました
- MoveFlow.java
  - ✓ AddressStateの更新(Move)に関するFlowを作成しました

※住所の更新=引っ越しなのでMoveと名付けました

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

既存のファイルの変更(要望1,2)

- IOUIssueFlow.java
  - ✓AddressStateをTransactionBuilderを用いて追加しました。

- IOUContract.java
  - ✓IOUのissuerとAddressStateのissuerが等しいかどうか確認する制約を追加
  - ✓IOUContractのIDとAddressStateのIDが等しいかどうか確認

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

既存のファイルの変更(要望2)

- IOUTransferFlow.java
  - ✓ AddressStateをTransactionBuilderを用いて追加しました。
    - ➢ 要望2を調査する際に、別ノードにトランザクションをTransferするためIOUTransferFlowにもRef.Stateに関する記述が必要になります

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

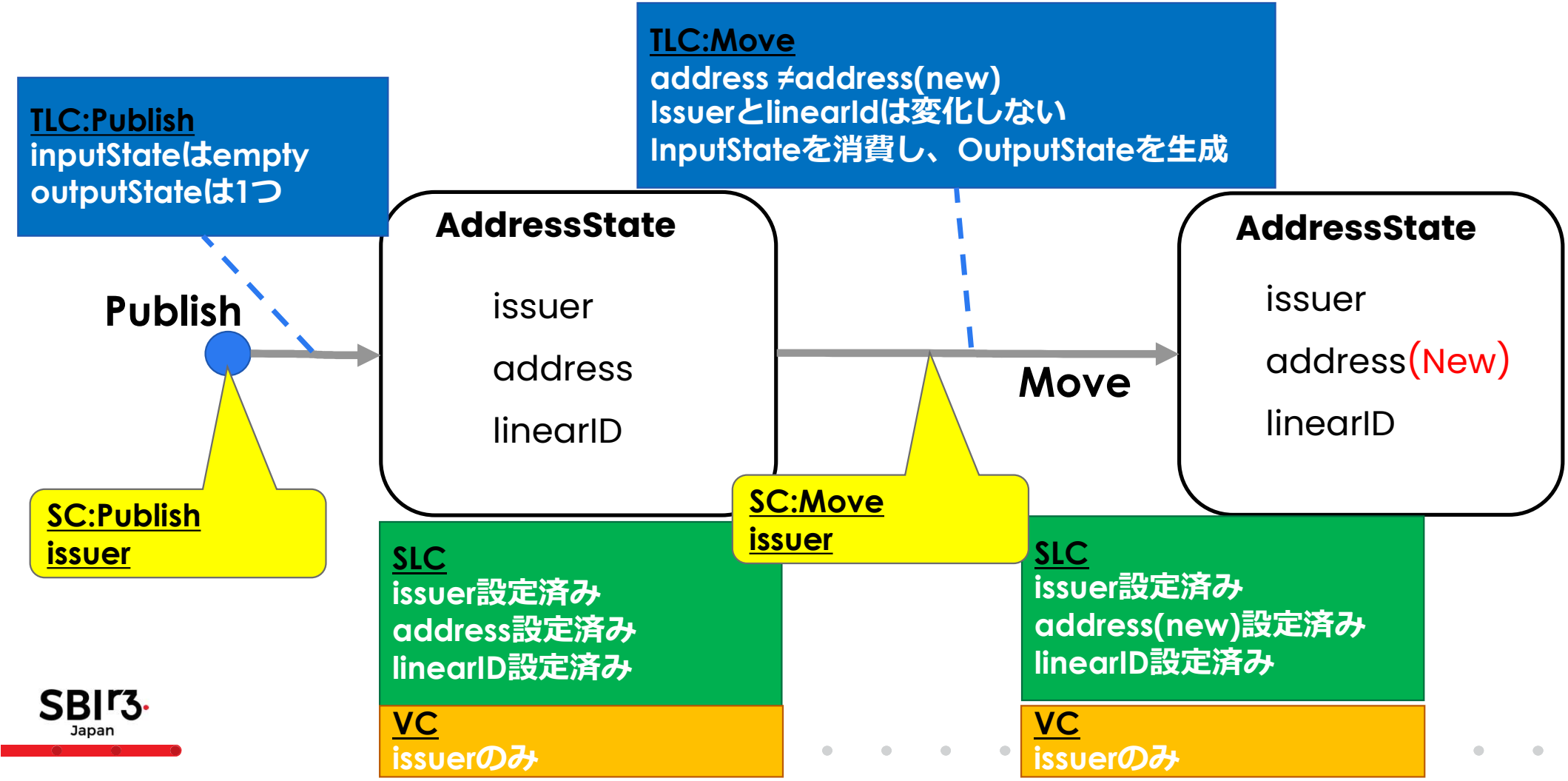
既存のファイルの変更(要望3)

- IOUIssueFlow.java
  - ✓基本的には要望1,2のコードと同じですが、一つ前のRef.Stateを探す処理を追加しました。
    - ➢ VaultQueryを用いて１つ前のRef.Stateのhashを探す処理
    - ➢ 探した１つ前のRef.Stateのhashを引数に取り、StateAndRefやQueryCriteriaを用いて一致するhash探す関数

SBI r3 Japan

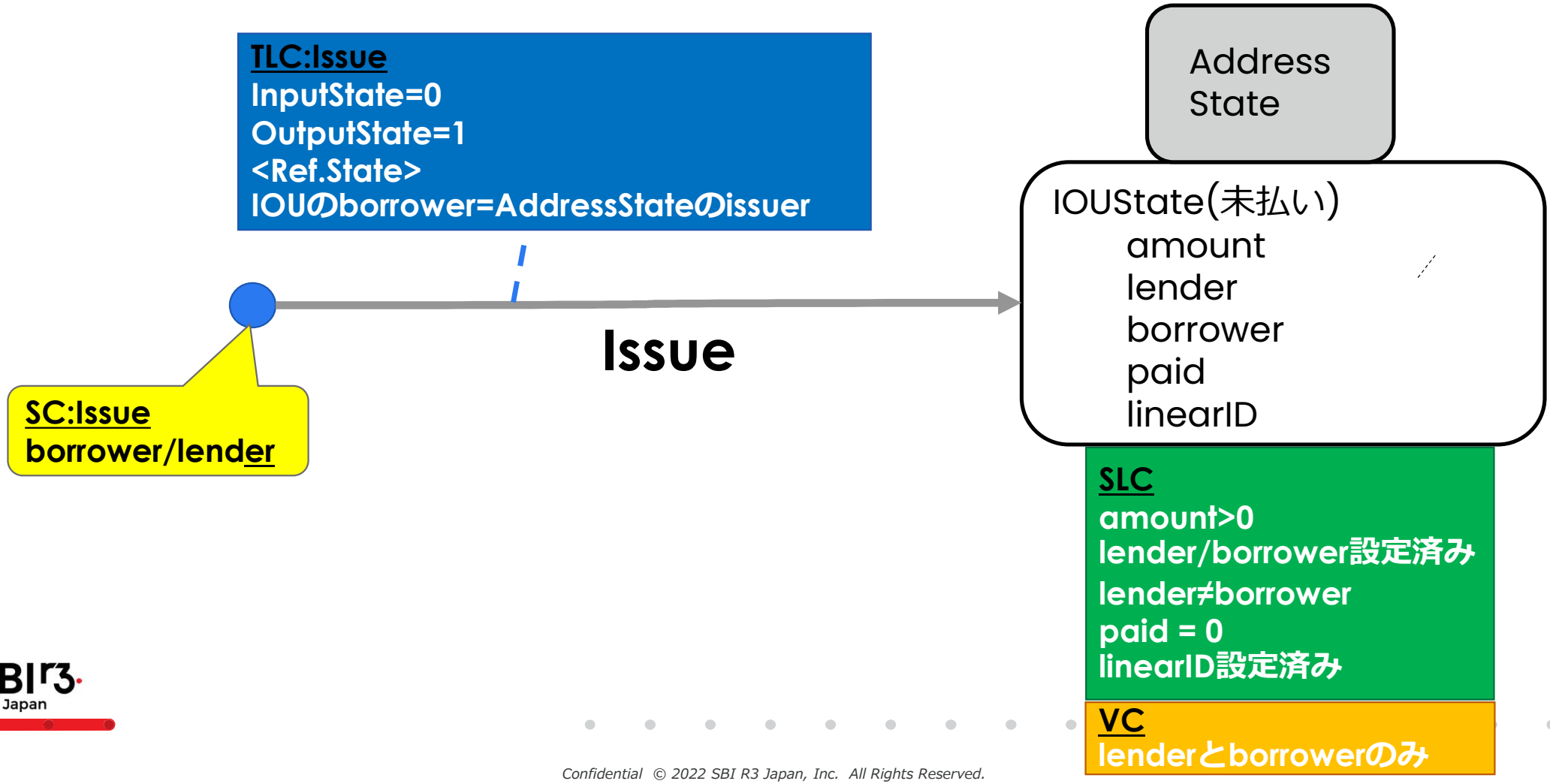
Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

CDL ~Ref.State~



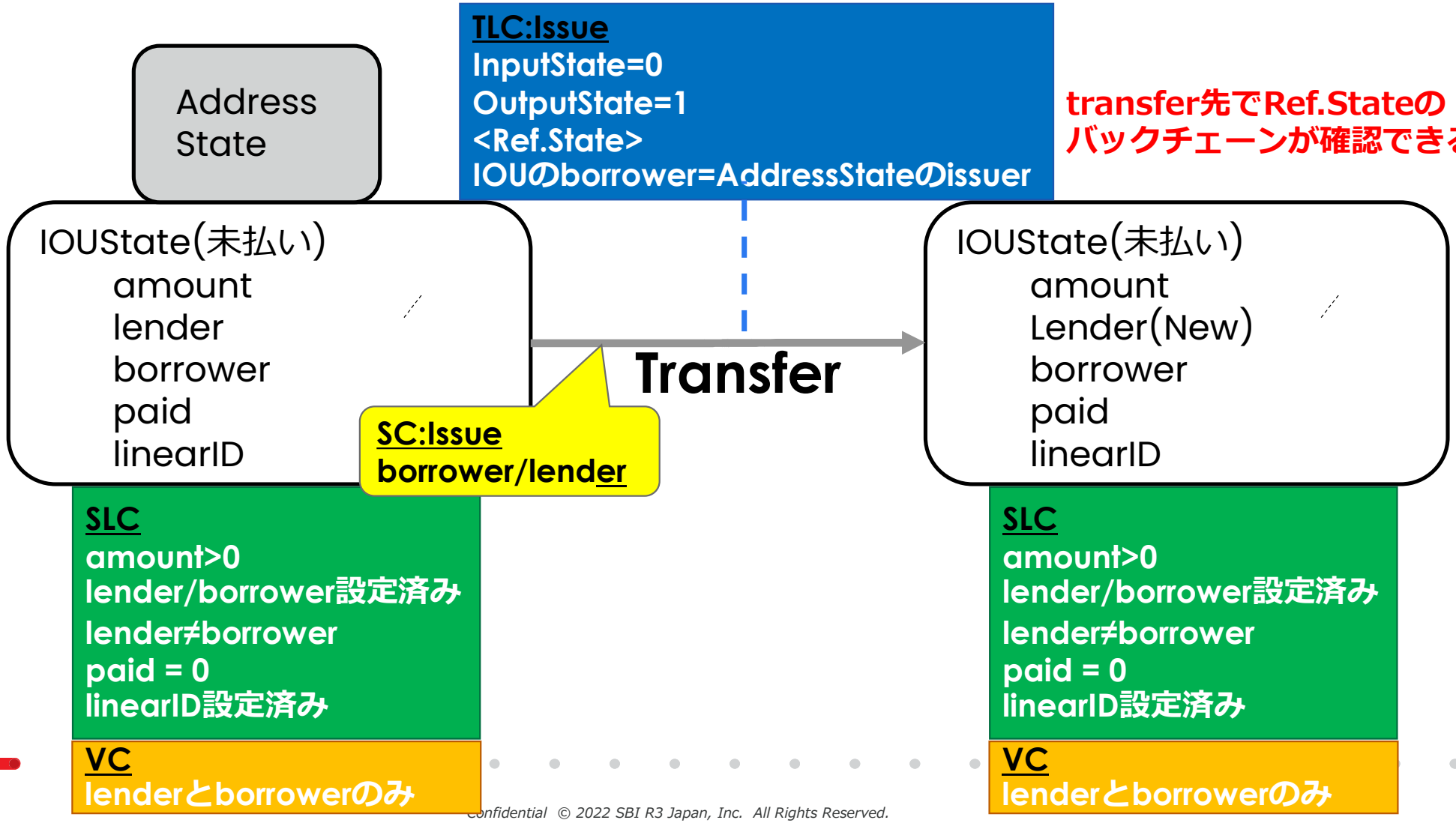
Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

CDL ~IOU Issue~



Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

CDL ~IOU Transfer~



Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

SBIr3 Japan

コード解説

調査内容(再掲)

- 1. Ref.Stateをトランザクションの中に含めるか？
  また、通常のコントラクトの中で検証できるか？

- 2. あるノードが発行したRef.Stateのバックチェーンを
  別のノードで確認できるのか？

- 3. 一つ前のRef.Stateをトランザクションに含められるか?

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

AddressState.java

・フィールド宣言

```
private static int ID_AddressState=1;
@NotNull
private final Party issuer;
@NotNull
private final String address;
@NotNull
private final UniqueIdentifier linearId;
```

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

AddressContract.java(Publishに関する制約)

- Inputに関する制約

```
//1. About InputState
require.using( $receiver: "No Address InputState should be consumed when publishing Address State",
        tx.inputsOfType(AddressState.class).isEmpty());
```

- Outputに関する制約

```
//2. About OutputState
final List<AddressState> outList=tx.outputsOfType(AddressState.class);
require.using( $receiver: "Only one OutputState should be created",
        expr: outList.size()==1);
```

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

AddressContract.java(Moveに関する制約)

- コードの簡素化のため

```
List<AddressState> inputState=tx.inputsOfType(AddressState.class);
List<AddressState> outputState=tx.outputsOfType(AddressState.class);
```

- Move特有の制約

```
//2. About Address information
require.using( $receiver: "Address information should be changed in Move transaction",
        !(inputState.get(0).getAddress()).equals(outputState.get(0).getAddress()));
```

```
//3. Other fields should not be changed.
require.using( $receiver: "Only address field should be changed in Move transaction",
        expr: inputState.get(0).getIssuer().equals(outputState.get(0).getIssuer()) &&
        inputState.get(0).getLinearId().equals(outputState.get(0).getLinearId()));
```

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

PublishFlow.java

- AddressStateの追加

```
//2. Add outputState and Command into TX.
AddressState state=new AddressState(getServiceHub().getMyInfo().getLegalIdentities().get(0), address);
Command txCommand=new Command(new AddressContract.Commands.Publish(),
        getServiceHub().getMyInfo().getLegalIdentities().get(0).getOwningKey());
TransactionBuilder txBuilder=new TransactionBuilder(notary)
        .addOutputState(state, AddressContract.ADDRESS_CONTRACT_ID)
        .addCommand(txCommand);
```

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

MoveFlow.java

- Inputとして用いるAddressStateの検索

```
//2. Find already published AddressState.
StateAndRef<AddressState> oldState=getServiceHub().getVaultService()
        .queryBy(AddressState.class).getStates().get(0);
AddressState oldStateData=oldState.getState().getData();
String newAddress=address;
AddressState newAddressState=new AddressState(
        oldStateData.getIssuer(),
        newAddress,
        oldStateData.getLinearId()
);
```

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

IOUContract.java

- ID_Contractフィールドの宣言

```
private int ID_Contract=1;
```

- AddressStateに関する制約

```
AddressState addressState =tx.referenceInputRefsOfType(AddressState.class).get(0).getState().getData();
require.using( $receiver: "The lender of IOUState and the issuer of AddressState should be matched."
        ,addressState.getIssuer().equals(outputState.getBorrower()));

//ID constraints
require.using( $receiver: "ID_AddressState and ID_Contract must be same."
        , expr: ID_Contract==AddressState.getID_AddressState());
```

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

IOUIssueFlow.java(要望1,2)

- Issuer(AddressState)からAddressStateを探す関数

```
public StateAndRef<AddressState> getAddressIssuer(Party addressStateIssuer){
    Predicate<StateAndRef<AddressState>> byIssuer=addressISU
            ->(addressISU.getState().getData().getIssuer().equals(addressStateIssuer));
    List<StateAndRef<AddressState>> addressLists = getServiceHub().getVaultService().queryBy(AddressState.class)
            .getStates().stream().filter(byIssuer).collect(Collectors.toList());
    if(addressLists.isEmpty()){
        return null;
    }else{
        return addressLists.get(0);
    }
}
```

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

IOUIssueFlow.java(要望3)

- VaultQuery等を使ってpreviousHashを探す

```
//1. get latest AddressState's hash by using vaultQuery.
QueryCriteria queryCriteria=new QueryCriteria.VaultQueryCriteria(Vault.StateStatus.ALL);
Vault.Page results =getServiceHub().getVaultService().queryBy(AddressState.class,queryCriteria);
//The index is set to 1 to see the item of "ref".
StateAndRef included_addressState=(StateAndRef) results.getStates().get(1);
SecureHash vault_addressHash=included_addressState.getRef().getTxhash();
```

```
//2. get previous AddressState's hash by using validatedTransaction.
StateRef results1 =getServiceHub().getValidatedTransactions().getTransaction(vault_addressHash).getInputs().get(0);
SecureHash previousHash=results1.getTxhash();
//Output previous hash to confirm.
System.out.println("previousHash="+previousHash);
```

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

IOUIssueFlow.java(要望3)

- 1つ前のAddressStateを探す関数

```
public StateAndRef<AddressState> getAddressContent(SecureHash previousHash){
    QueryCriteria queryCriteria=new QueryCriteria.VaultQueryCriteria(Vault.StateStatus.ALL);
    //search the hash equal to the previous hash and put it in the list.
    Predicate<StateAndRef<AddressState>> byHash=address_hash
            ->(address_hash.getRef().getTxhash().equals(previousHash));
    List<StateAndRef<AddressState>> addressHashLists=getServiceHub().getVaultService()
            .queryBy(AddressState.class,queryCriteria)
            .getStates().stream().filter(byHash).collect(Collectors.toList());
    //Output previous hash to confirm.
    System.out.println("the content of list="+ addressHashLists);
    if(addressHashLists.isEmpty()){ return null; }
    else{ return addressHashLists.get(0); }
}
```

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

SBIr3 Japan

動作デモンストレーション

調査内容(再掲)

- 1. Ref.Stateをトランザクションの中に含めるか？
  また、通常のコントラクトの中で検証できるか？

- 2. あるノードが発行したRef.Stateのバックチェーンを
  別のノードで確認できるのか？

- 3. １つ前のRef.Stateをトランザクションに含められるか?

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

前提条件

- コマンドプロンプト
  - ✓ cd サンプルプログラムのルートディレクトリ
  - ✓ gradlew deployNodes
  - ✓ cd build
  - ✓ cd nodes
  - ✓ runnodes

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

SBIr3 Japan

要望3のエラーについて

調査内容(再掲)

- 1. Ref.Stateをトランザクションの中に含めるか？
  また、通常のコントラクトの中で検証できるか？

- 2. あるノードが発行したRef.Stateのバックチェーンを
  別のノードで確認できるのか？

- 3. １つ前のRef.Stateをトランザクションに含められるか?

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

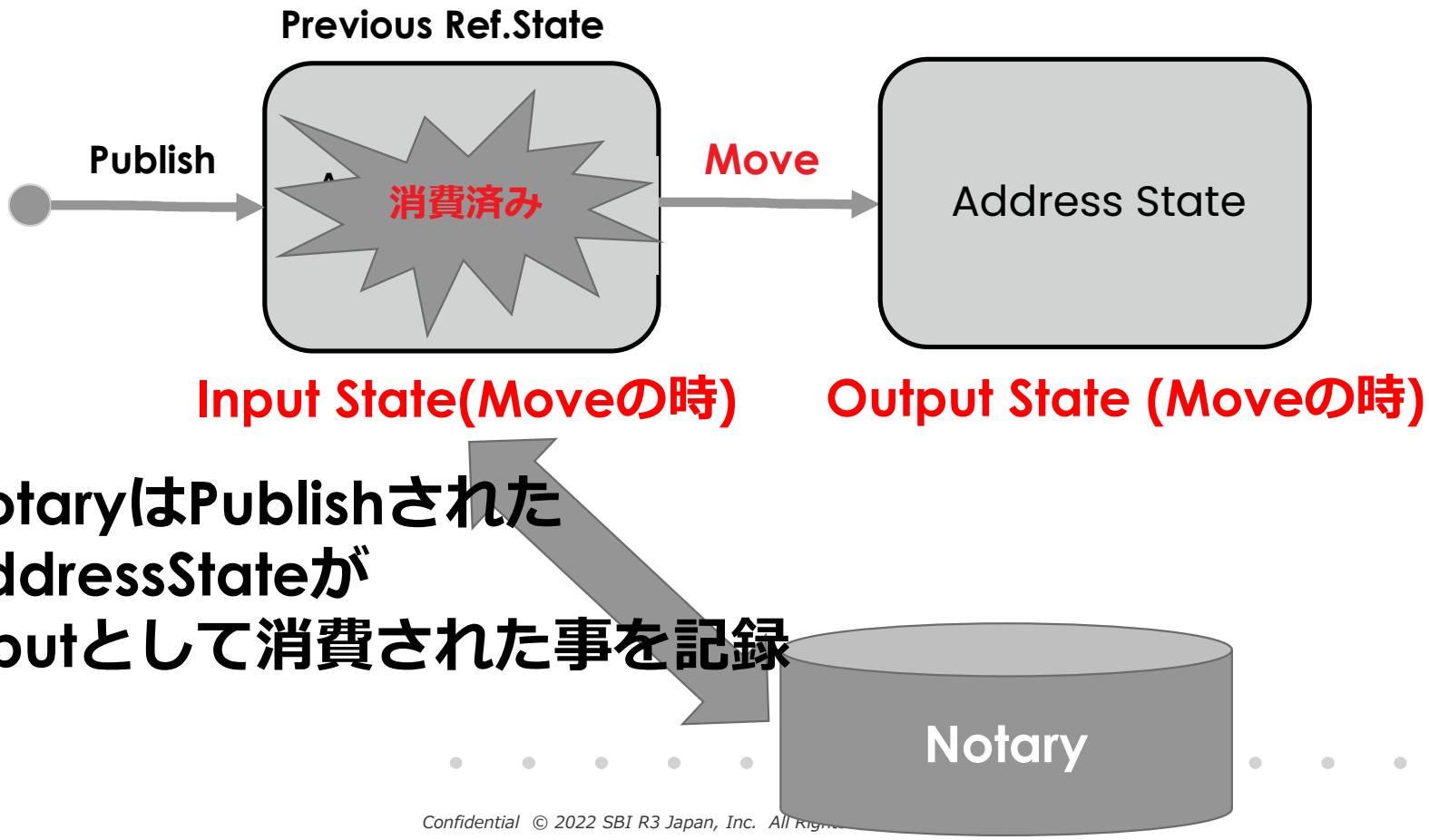
要望3が実現不可能な理由

- 要望3はサンプルコードを走らせると下記のエラーが表示されます
  - ✓ One or more input states or referenced states have already been used as input states in other transactions.
- Moveする段階で最初にPublishしたPrevious Ref.Stateに該当するAddressStateがInput Stateとして使われたことをNotaryが記録するため、含めることができない

SBI R3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

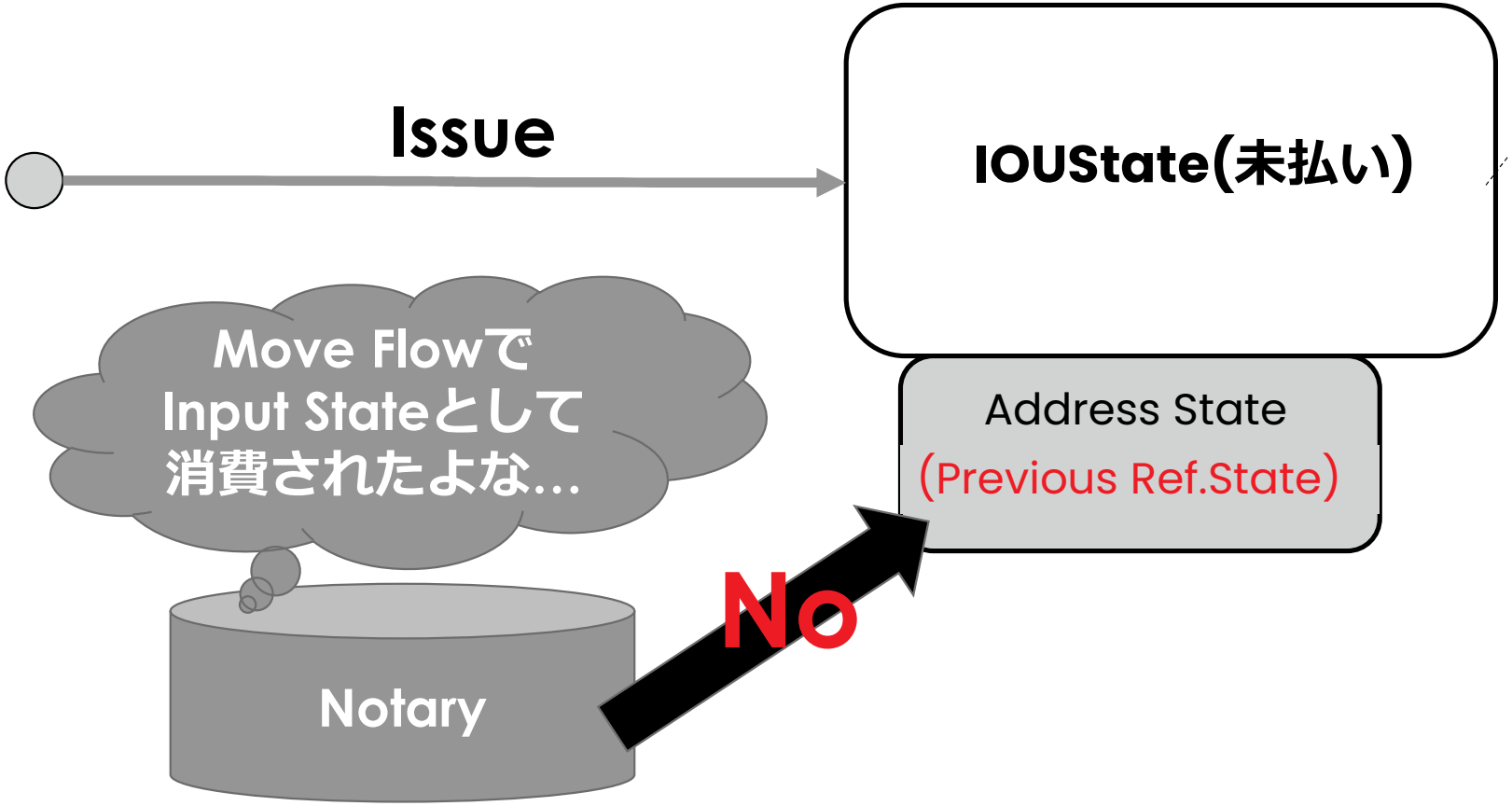
要望3が実現不可能な理由



NotaryはPublishされた
AddressStateが
inputとして消費された事を記録

Confidential © 2022 SBI R3 Japan, Inc. All Rights

要望3が実現不可能な理由



Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

作成したファイルの場所

- 以下URLのgithub上からダウンロードできます。
- 要望1,2に対応するコード
  - ✓ https://github.com/ShotaIMO/investigation-of-ReferenceState
- 要望3に対応するコード
  - ✓ https://github.com/ShotaIMO/investigation-of-Previous-ReferenceState

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

参考文献

- https://github.com/corda/samples/tree/97065e34c513379262bbf0e43bc5eb0e1d0a7212/reference-states

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

おわりに

- 調査をする上で協力していただいたSBI R3 Japan
  プロダクトサービス部の皆様、ありがとうございました。

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.

SBI r3 Japan

Confidential © 2022 SBI R3 Japan, Inc. All Rights Reserved.