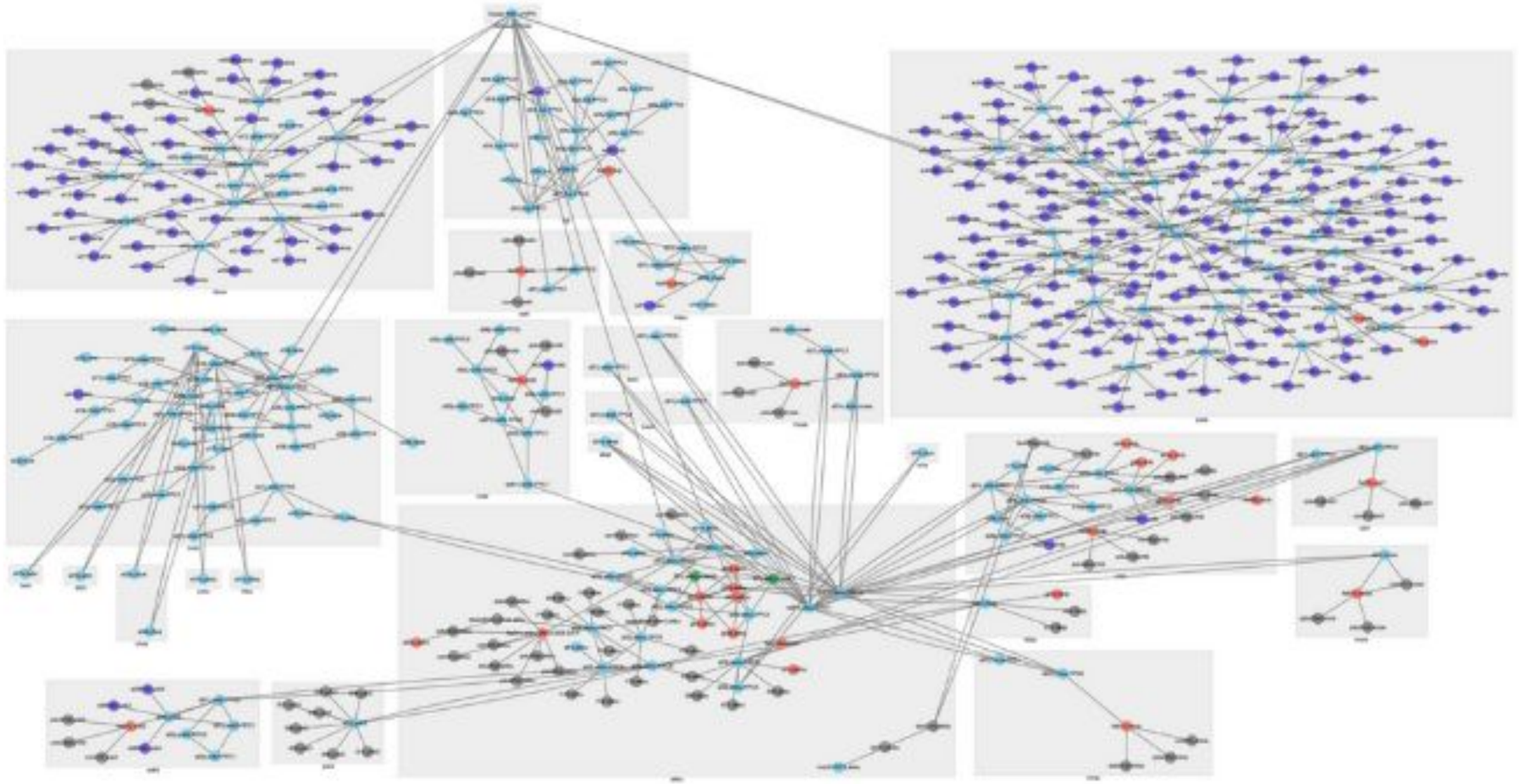
NSOのトランザクション内に SO開通テストを組み込もうとしたが難しかった

2024年2月28日

NTTコミュニケーションズ株式会社
坂井 立晟

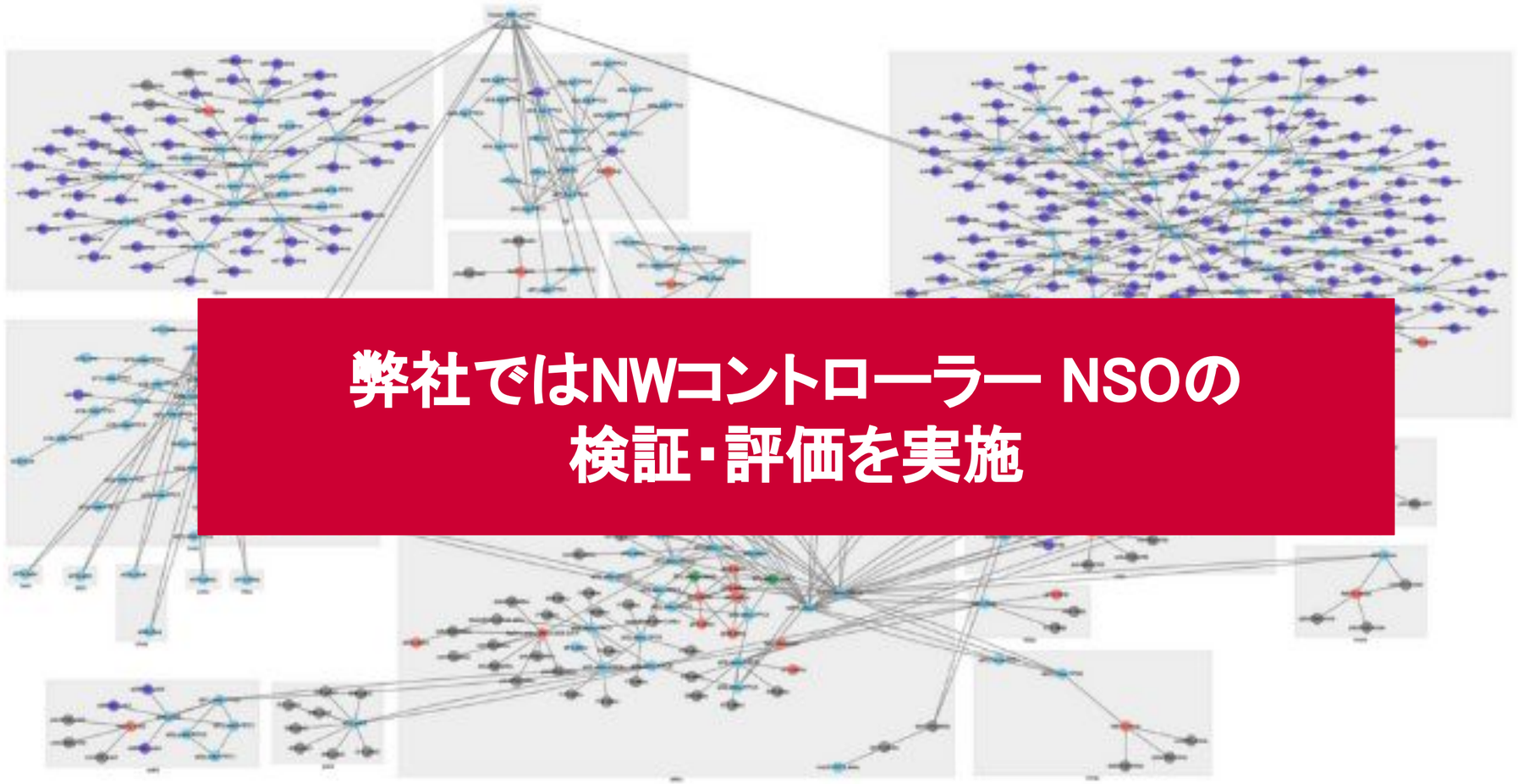
手動でのNWサービスの構築、運用は大変

- 以下は社内検証網のNW図
- **手動でのNWの構築、運用は大変**



手動でのNWサービスの構築、運用は大変

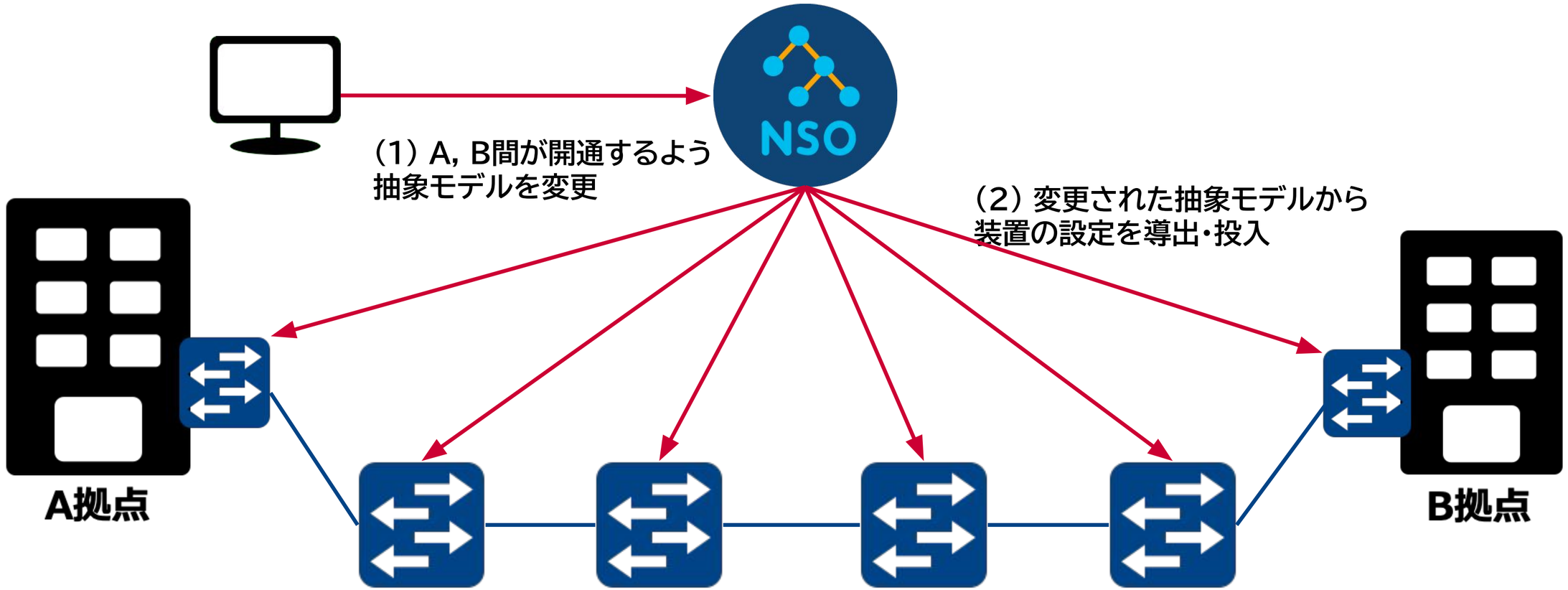
- 以下は社内検証網のNW図
- **手動でのNWの構築、運用は大変**



弊社ではNWコントローラー NSOの
検証・評価を実施

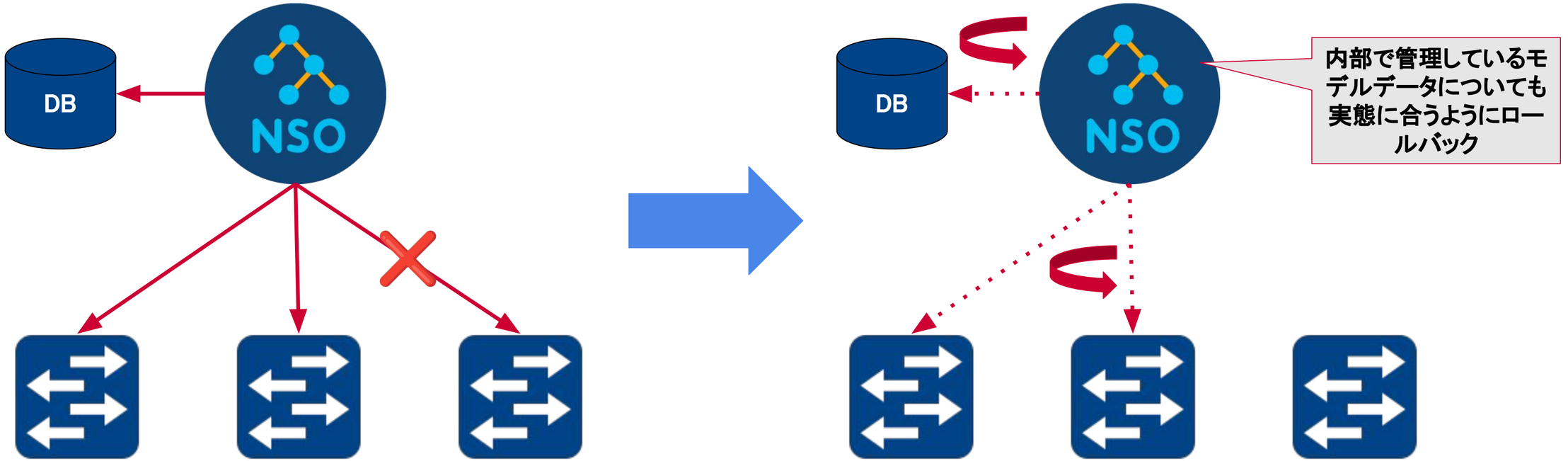
NSOとは何か？

- NWサービスを**抽象モデル化・宣言的記述しプログラム制御することに特化**したModel Driven なNWコントローラー
- **ユーザーは各装置のコンフィグを直接意識することなく、NWサービスのデリバリが可能になる**



装置設定に失敗した時のロールバックを実装レスで提供

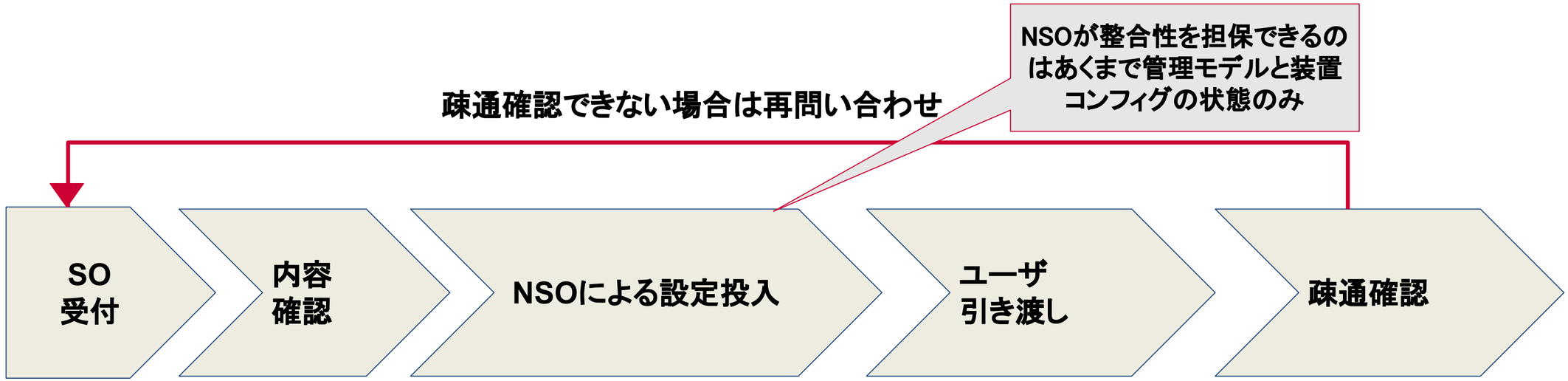
- NSOはユーザーからの管理モデル変更から装置設定投入までをトランザクションと呼ばれる論理単位で実行できる。
- **追加実装不要で管理している装置と対応するモデルデータの整合性を担保することができる。**



NSOトランザクション例

SO対応における疎通確認の自動化検討

- 以下のようなユーザー引き渡し後に疎通確認を実施するSO対応フローを考えていた。
- この場合NSOによる設定投入が正常に成功したとしても、物理故障等の要因から疎通が正常でないケースが存在する。
  - この状態に重ねてSO対応が入ってしまうとよりカオスな状態になっていく。
  - **オーダー受付から設定、疎通確認まで一連の処理を自動化したい。。。**



SO対応フロー例

実現したいこと

- SO対応後に疎通確認がNGの場合、装置管理データと実機コンフィグの整合性を確実に担保できるように切り戻しを実施したい。
- SO対応後にNWが正常でない状態になった場合、設定投入から出来るだけレスが少ない状態で切り戻しさせたい。

疎通確認、及び切り戻しをNSOトランザクションの中で実施できれば実現可能

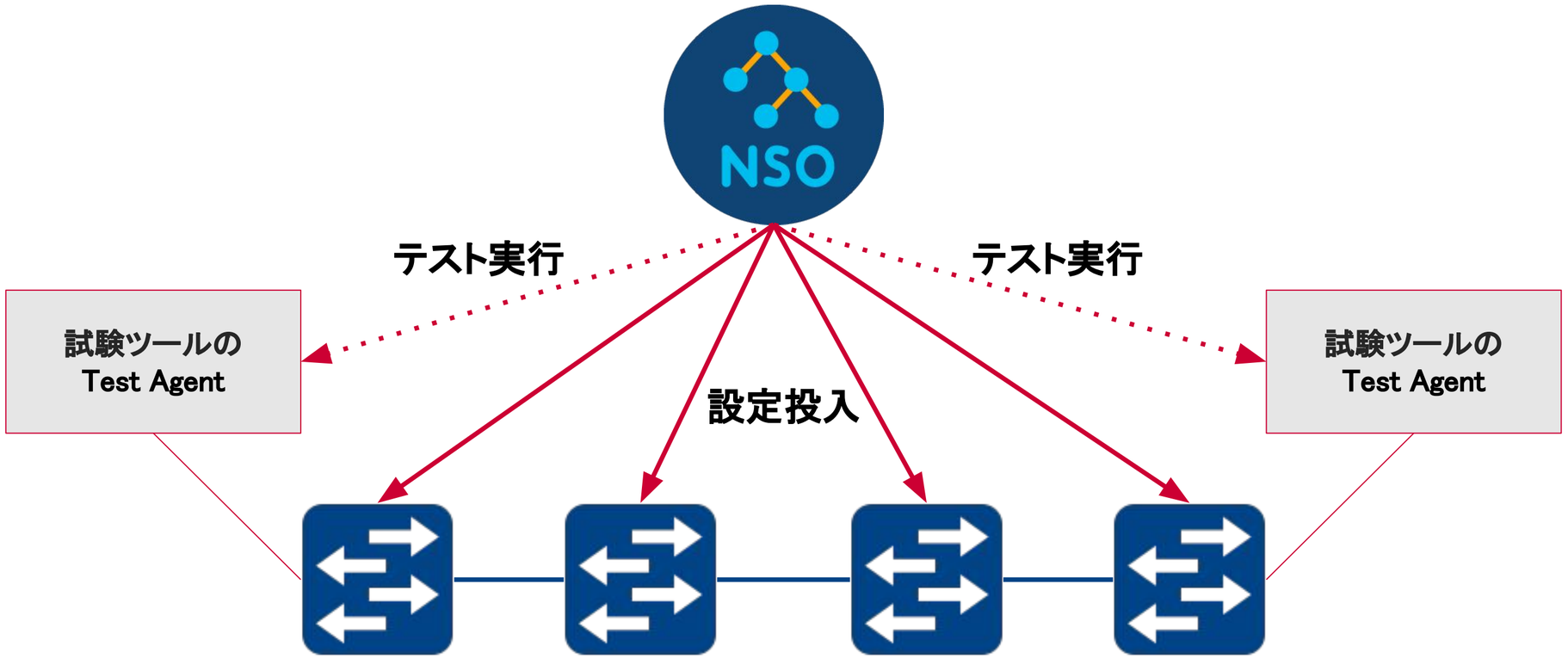
実現したいSO対応フロー

実現したいSO対応フローに向けた検証

- 以下3つの方式について検証・評価を実施した。
  - **検証手法1: 試験ツールをNW装置としてNSOから管理**
  - **検証手法2: NSO Kickerの活用**
  - **検証手法3: NSO Post-modificationの活用**

検証手法1 試験ツールをNW装置として管理

- 試験ツールをNW装置と同じように管理対象に含めて、SO投入時に試験ツールに関してはコンフィグ投入をテスト実行に置き換える。**テストが失敗した場合は設定失敗としてロールバック**。

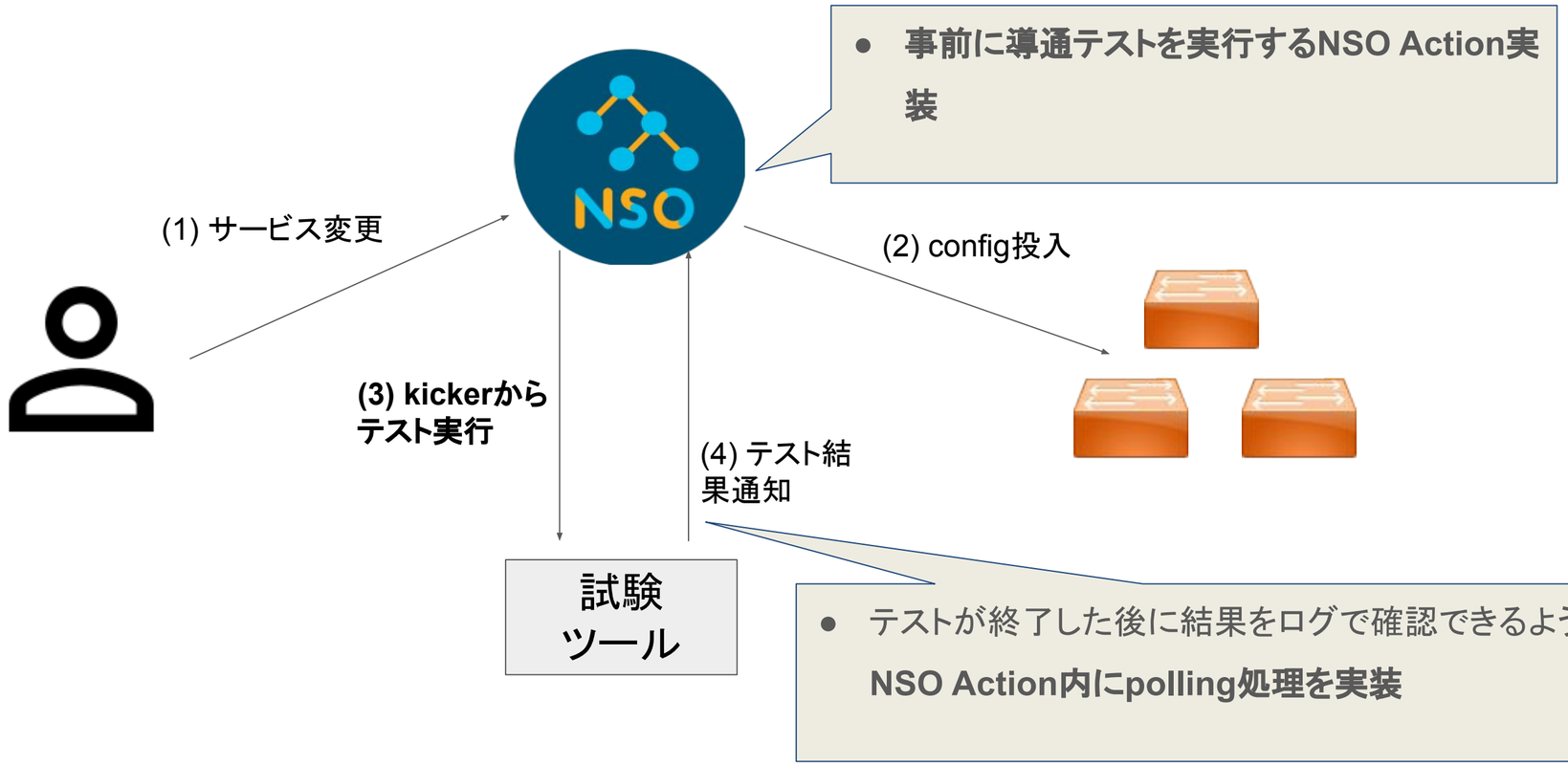


検証手法1の結果→上手くいかなかった

- 試験ツールのテスト制御パラメータがNSOから設定変更できなかった。
  - テスト実行に関する制御のパラメータが試験ツールのYANGで設定投入不可(config false)のパラメータとして指定されており、NSOから設定を投入することができず**実装を断念した**。
- （後にNSOの追加調査から、テスト実行処理を設定パラメーターで扱えていたとしても上手くいっていなかったことが判明、、、）

検証手法2 NSO Kickerの活用

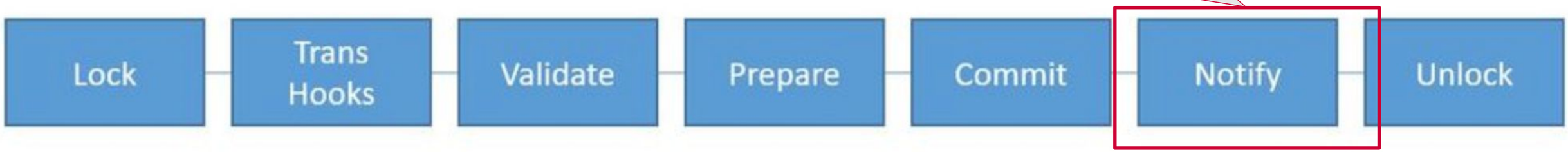
- NSO Kicker: NSOの装置への設定投入処理をトリガーにして事前に登録した処理(NSO Action)を起動する仕組み
- 以下のような構成で、設定投入時にテストが自動実行されるように実装



検証手法2の結果→上手くいかなかった

- NSOのトランザクション処理でのNotify部分で発火するが、この時点でcommit(装置への設定投入の確定)が終わっている → **Kickerが発火した設定投入のトランザクション内でロールバックさせることができない**
- (optional) 現状の実装では、テスト結果を確認するために、NSOが吐き出すログを毎回確認する必要があり使い勝手が悪い。。。

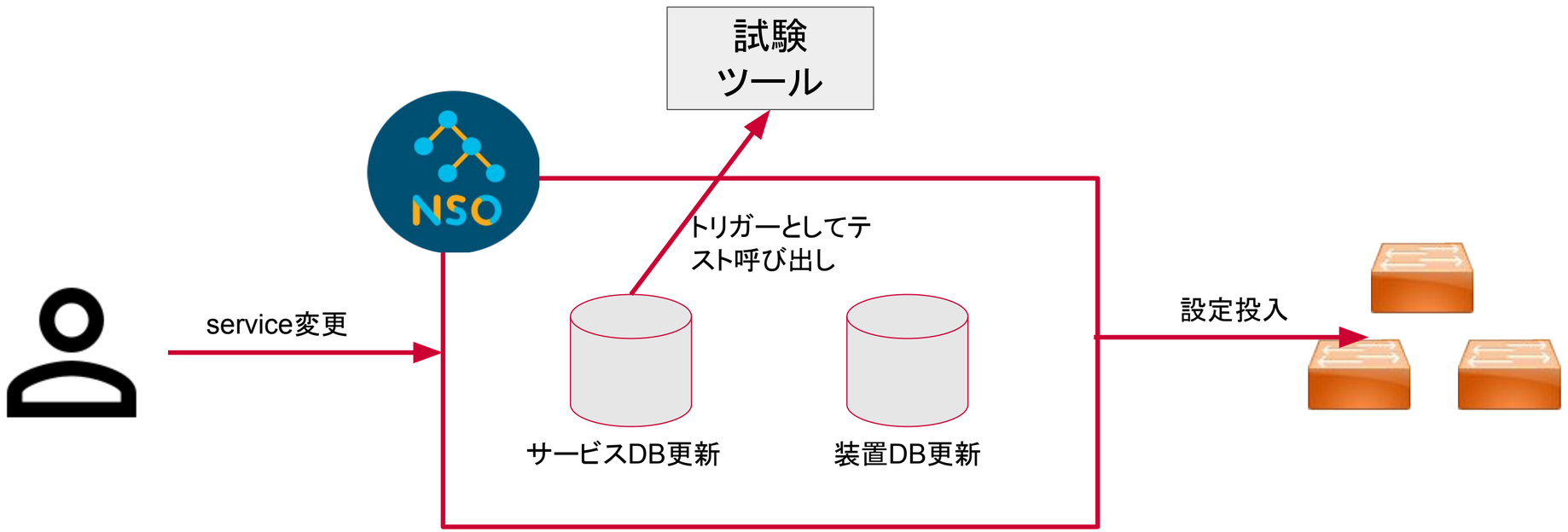
実際にKickerを動作させると、このNotify部分で
テストを実行していた
→ **commit内で装置情報が確定した後なので、設定と同トランザクション内で切り戻しは出来ない。**



NSOのトランザクション処理流れ

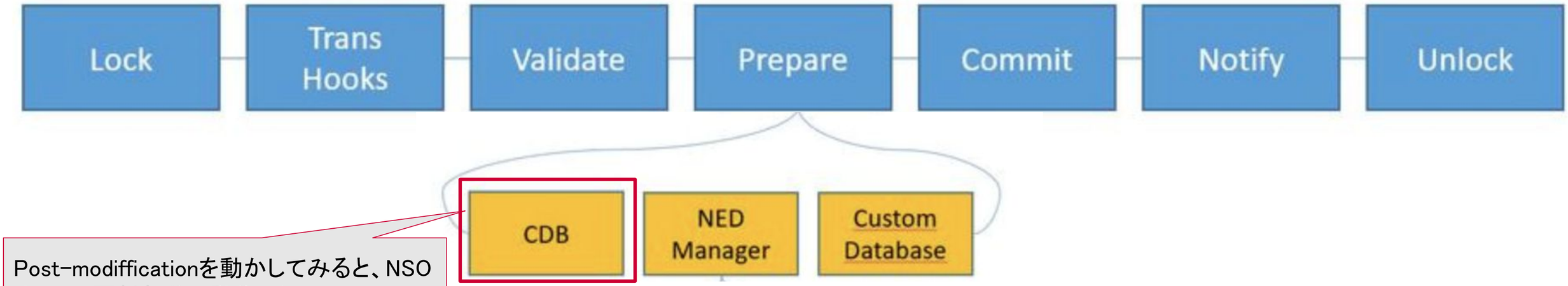
検証手法3 NSO Post-modificationの活用

- NSO Post-modification: 指定した管理モデル(サービス、装置)の変更をトリガーとして処理を実行する
- 以下のように、指定したサービスモデルが更新された際に導通テストを実行するように実装



検証手法3の結果→上手くいかなかった

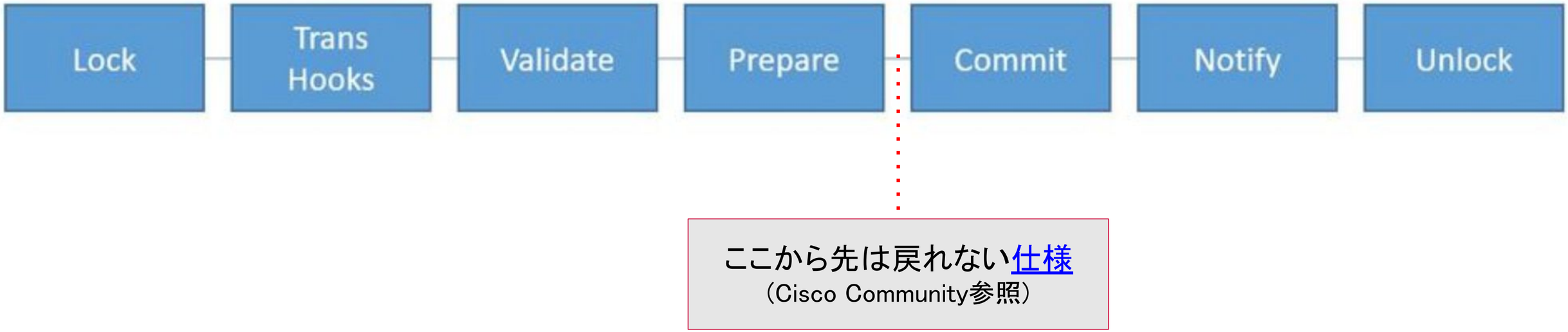
- 装置に設定が投入される前に導通テストが走ってしまう
  - トランザクション内で管理しているモデルデータの変更を装置投入前に実施しているのが原因(下フローのprepare内)
  - モデル変更をトリガーにするPost-modificationの仕様的に設定後の導通テストを入れ込むのは無理そう、、、。



Post-modificationを動かしてみると、NSO内のDB更新部分で動作した
→ **装置設定は反映されていないため、疎通試験を行うことができない**

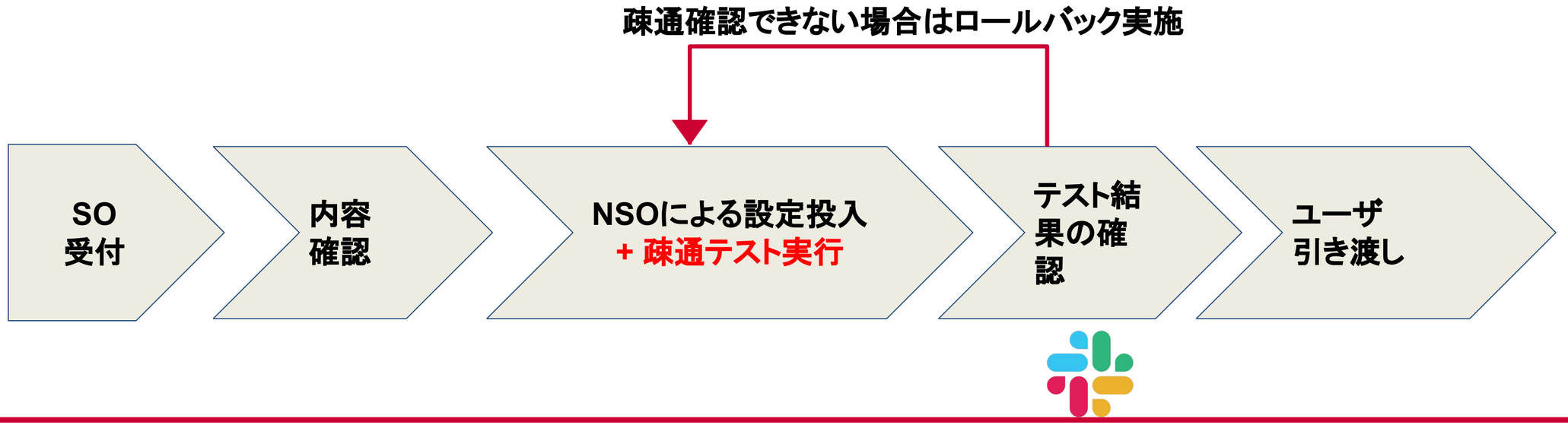
そもそも装置設定投入後にトランザクション内でロールバックさせることは可能なのか?

- 結論: NSOが設定投入 = commitした時点で切り戻しを想定しないため実現は難しい
  - Prepare 時点でも装置に変更が書き込まれてはいるが、**適用はされていない状態**なので、投入した設定で疎通テストを実行させるのは無理



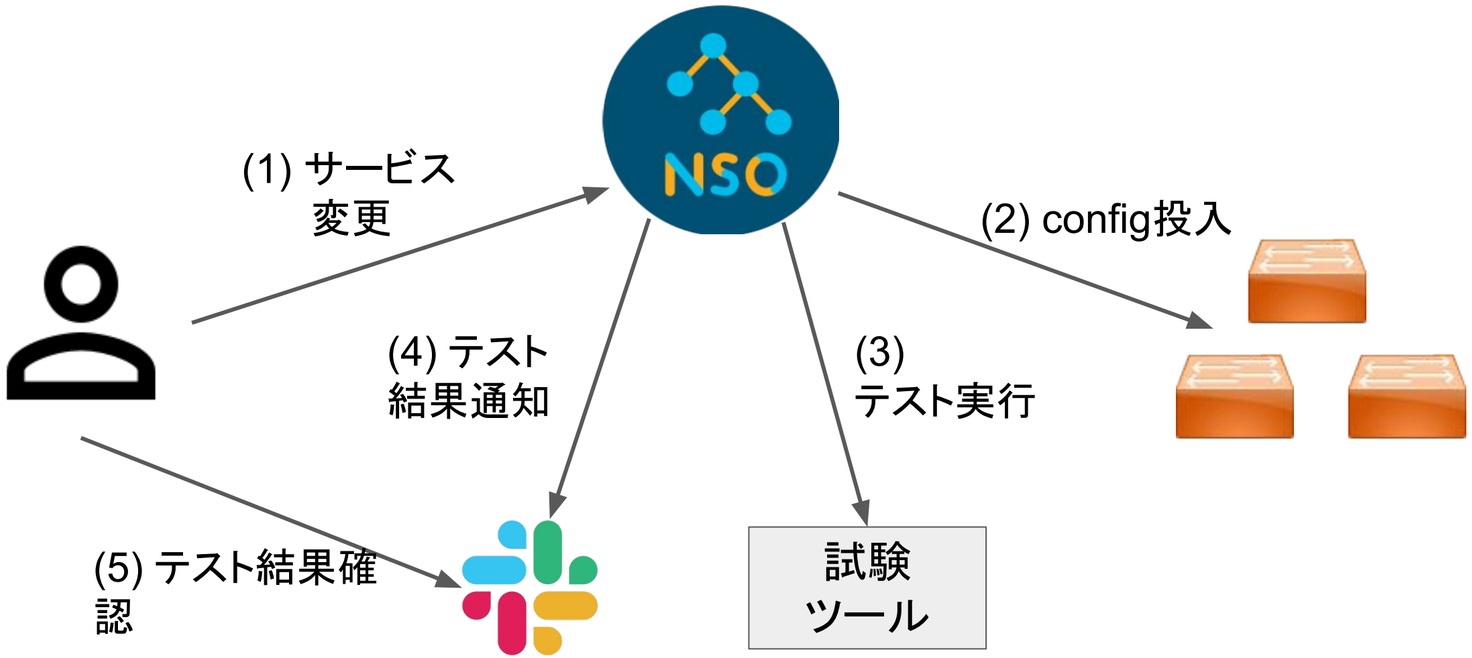
実現したいことを再検討

- 当初やりたかったNSOトランザクション内での疎通試験・切り戻し実施は難しいことがわかったので検証するSOフローを以下のように変更
  - **NSOで設定投入直後に疎通テストを自動実行し、**結果を確認して適宜ロールバックさせる
  - この構成でもユーザー引き渡し前に疎通確認を(自動で)実施できるようになり、予期しない疎通状態のNWにSOを重ねてしまう事態を防ぐことができる。
- **前の検証結果からKicker使えばできそうだがテスト結果の確認が大変だった → Slack連携の検討**



NSO KickerとSlack連携によるテスト自動化

- 以前検証していた構成だと、テスト結果を確認するためにはNSOが内部で持っている実行ログを参照する必要があったが、**Slackにテスト結果を通知し、ログファイルの確認作業を不要にした。**
- 合わせてkickerの実行結果からトランザクションIDと変更された装置を取得して記載することで、**時間が経ってから見返しても自分の作業と疎通テスト結果が紐づけられるようになった。**



まとめ・議論したいこと

- NSOトランザクション内での疎通試験・切り戻しを実現するため3つの手法を検証した。
  - どれも設定投入後にNSOトランザクション内での切り戻しを実現することができなかった。
  - **そもそもNSOの仕様上、装置の設定反映=commit後にロールバックできないことがわかった。**
- 目標設定を見直し、NSO Kickerを用いて、導通試験の実行までを自動化することができた
  - Kickerの実行結果を確認するためにはNSOの実行ログを確認する必要があったため、Slackと連携
  - **当初の目標とは異なるものの、ユーザー引き渡し前の疎通確認をNSO Kicker(+Slack)によって自動化することができた。**
- 本取り組みに対してのコメントや、NSOトランザクションについて何か知見あれば
  ぜひご教授をよろしくお願い致します🙇