トップエスイー修了制作

RESTful Web API 設計フロー

末永　晋也

開発における問題点

背景　近年の多様化するWebサービスでは、従来のブラウザ向けのインタフェースに加え、RESTful Web APIによる提供も一般化している。

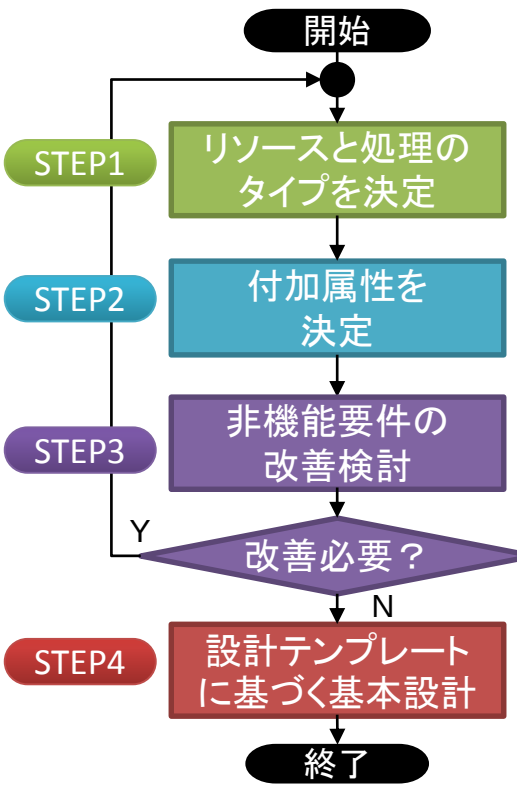
課題　RESTful Web API 設計時の選択と、非機能要件との対応が把握できず、ユースケースに適さないWeb API 設計となる危険がある。

手法・ツールの提案による解決

ユースケースに適した RESTful Web API を簡単に設計できるフローを提案する。

- 設計時の選択肢をメニュー化し、選択によって変化する非機能要件を明確にした、RESTful Web API 設計フローを提案する。
- フローでの選択に対応した典型的な設計テンプレートを提供する。

非機能要件に注目した RESTful Web API 設計フロー



STEP1

リソース3タイプと処理9タイプから選択し、基本的なリクエストレスポンスを決定する。選択を補助する各タイプの選択基準と、相対的な非機能要件を明確化。

非機能要件対応の例

リソース	処理	メソッド	標準IF	耐障害性	応答速度	拡張性
モノ	取得	GET	○	○○	○	○
	削除	DELETE	○	○	△	△

STEP2

付加属性4タイプから選択し、複数のリクエストレスポンスで実現する非同期処理などの特性を決定する。選択を補助する各タイプの選択基準と、非機能要件の変化を明確化。

非機能要件変化の例

付加属性	標準IF	耐障害性	応答速度	拡張性
トランザクション	×	△	△	△

STEP3

改善ポイント3点を適用し、STEP1,2で明確化した非機能要件が改善可能かを検討する。適用を補助する各ポイントの適用条件と、改善の可能性がある非機能要件を明確化。

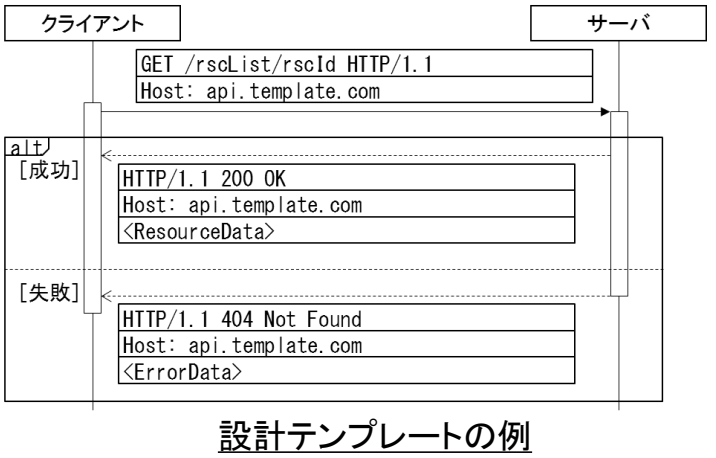
改善の可能性がある非機能要件の例

改善ポイント	標準IF	耐障害性	応答速度	拡張性
非同期クライアント利用	○	○	○	○

設計テンプレート

STEP4

STEP1～3の選択に対応した、設計テンプレートを元に基本設計を行う。
各リソースタイプ、処理タイプ、付加属性に対応した典型的なシーケンスをテンプレート化。



設計テンプレートの例

評価

ユースケースの異なる、チケット予約Webアプリと、制御監視システムの2件について提案設計フローを試験適用し、非機能要件に応じた設計が可能であることを確認した。

事例での非機能要件改善ポイント適用の例

- 非同期処理を、クライアントで実現する設計に変更し、各非機能要件を向上。
- ログ書込みの処理タイプを変更し、耐障害性を向上。

NII 国立情報学研究所 National Institute of Informatics
トップエスイー ～サイエンスによる知的ものづくり教育プログラム～