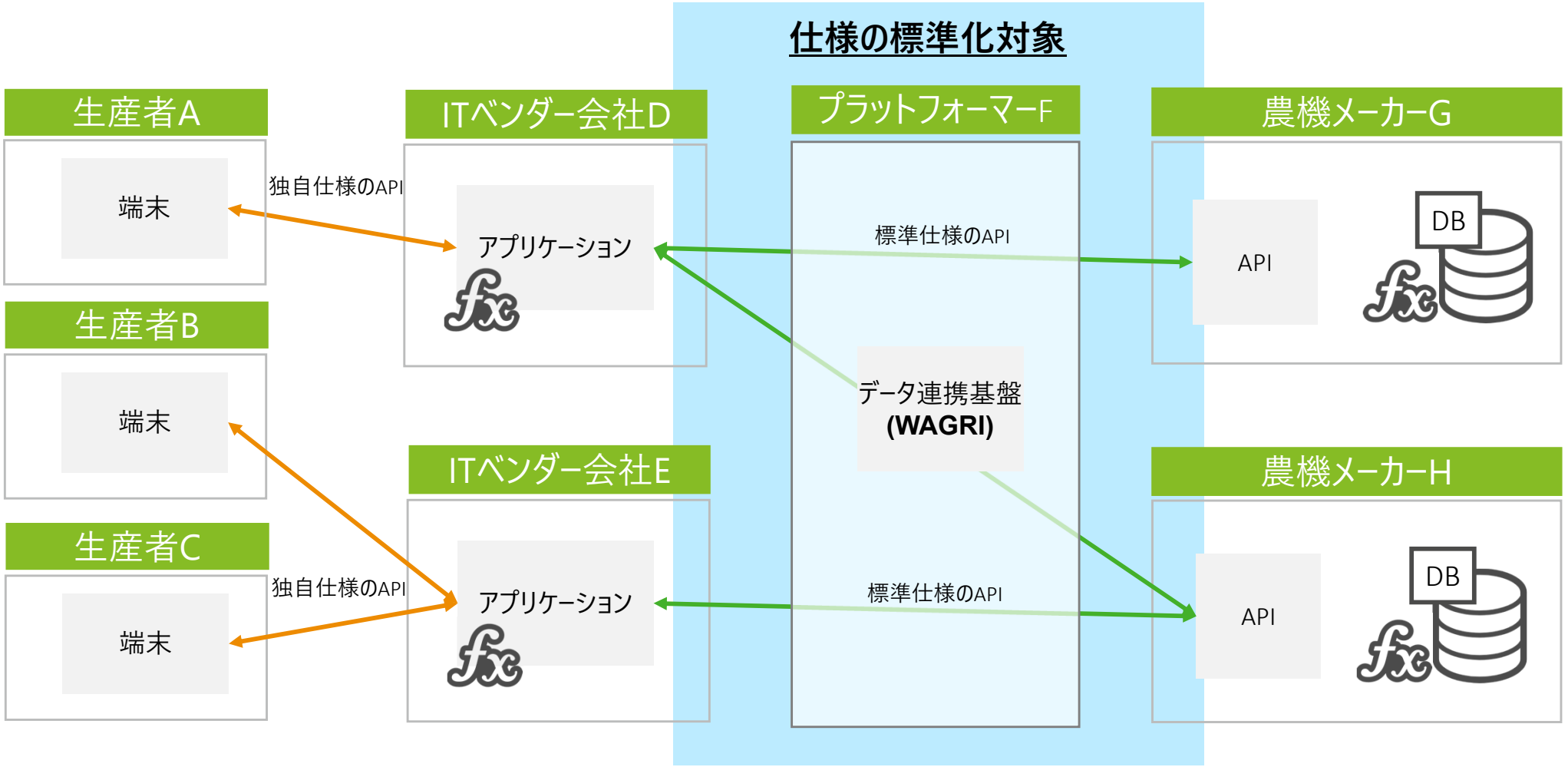
農機共通化ＡＰＩ仕様書作成の要点

農機共通化API仕様書作成の意義

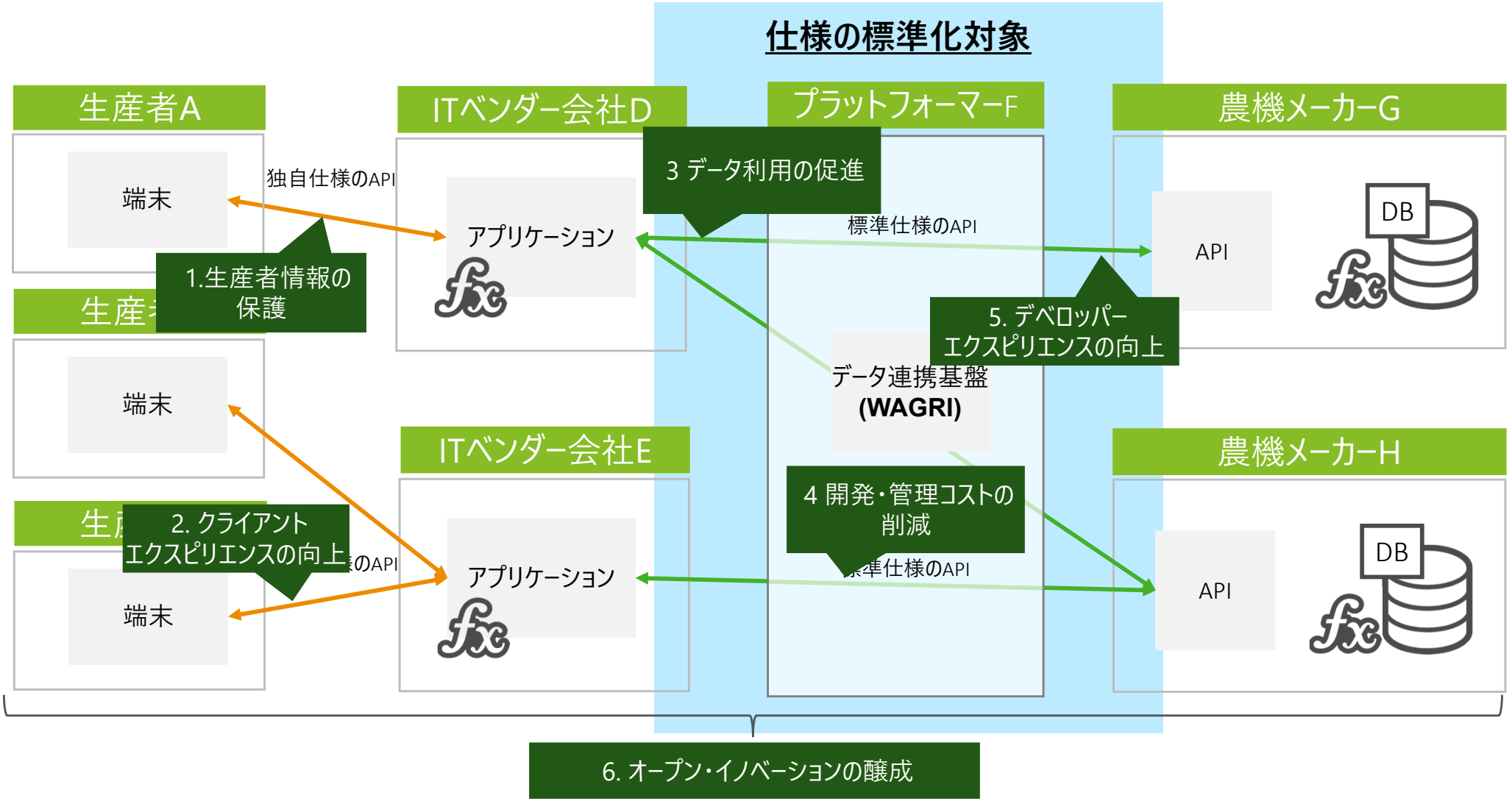
本業務のオープンAPI基盤には「生産者」「ITベンダー」「プラットフォーマー」「農機メーカー」の各プレイヤーが参画、このうち「ITベンダー」「プラットフォーマー」「農機メーカー」間のやりとり（API）を標準化していく

ステークホルダーの関連と標準化の位置付け



標準化により各プレイヤーへ様々なメリットを提供できる（1/2）

標準化によるメリット



標準化により各プレイヤーへ様々なメリットを提供できる（2/2）

標準化によるメリット詳細

No	タイトル	詳細	受益者
1	生産者情報の保護	• 認証・認可の仕組を標準化することで生産者情報のセキュアーな提供を実現する • ITベンダーは信頼性の高いシステムを提供可能となり、生産者は自身の情報保護が可能となる	生産者 ITベンダー社
2	クライアント エクスピリエンスの向上	• 標準化により各APIのパフォーマンス・信頼性等が平準化されるため生産者としての利便性向上が実現する	生産者
3	データ利用の促進	• 標準化によりAPIにおける各種定義の一貫性が確保できるため、ITベンダーはデータの統合・加工・集計が容易になる。 • 農機メーカー・プラットフォーマーは提供データの利用者増加が期待できる	農機メーカー ITベンダー社 プラットフォーマー
4	開発・管理コストの 削減	• 標準化によるシンプルなAPIにより提供側（農機メーカー）と利用側（ITベンダー）に発生する開発・管理コストを最小化できる	農機メーカー ITベンダー社
5	デベロッパー エクスピリエンスの向上	• 標準化を通じた理解しやすい仕様の提供によってAPIの開発・導入・統合が合理化され、デベロッパー（農機メーカ・ITベンダー）にとっての利便性が向上する • プラットフォーマーにとってはデベロッパーの参加を促進できる	農機メーカー ITベンダー社 プラットフォーマー
6	オープン・イノベーション の醸成	• 標準化によりAPI利用に向けたハードルが下がり、各プレイヤーの参加・及び連携が促進されることで新しいサービスを迅速に開発するための基盤が整備される	生産者 農機メーカー ITベンダー社 プラットフォーマー

農機共通API仕様書（関連文書等を含む）の構成案

農機共通API仕様書の構成案は下記のとおり

農機共通API仕様書の構成案

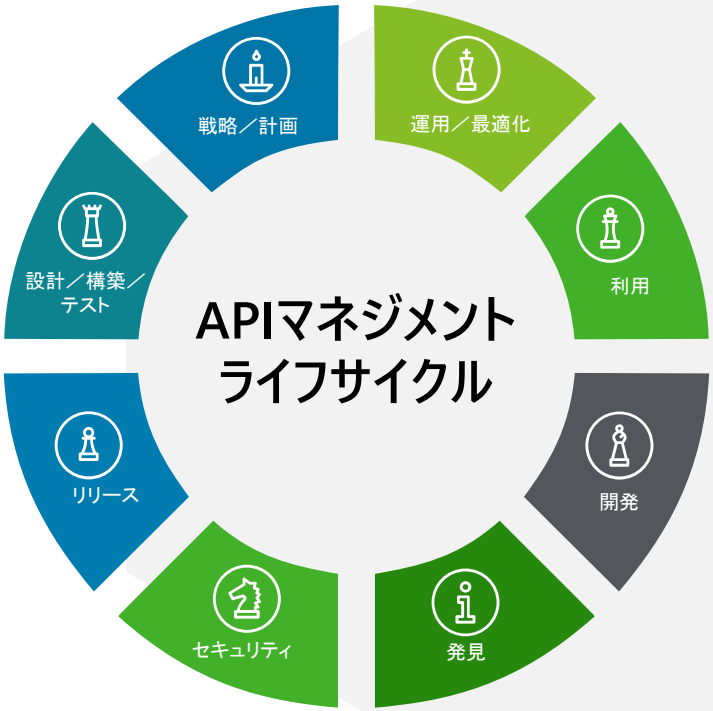
- 農機共通API仕様書の目的と位置付け
  - APIライフサイクルにおける適応範囲

- 農機共通API仕様書の構成
  - API仕様の原則（API 開発・仕様決定で留意すべきハイレベルの概念）
  - API開発標準（アーキテクチャ、データ表現、認証・認可等の基本的仕様）
  - API定義（データ項目、定義や構造、エンドポイント、パラメータ等）

1. APIライフサイクルと共通API仕様書の適応範囲

仕様書の適応範囲

農機メーカの独自対応



領域と分担	内容
戦略/計画	■ API活用の戦略の明確化 ■ ロードマップ作成における評価軸、アーキテクチャおよびアーキテクチャガイドラインの策定
設計	■ APIアーキテクチャに沿った具体的なAPI設計 ■ OpenAPI Spec (Swagger)の作成 ■ 不正利用の余地の少ないAPI設計
構築/テスト	■ API環境の構築、テスト
セキュリティ	■ プラットフォームの安全性の検討やその担保
開発	■ OpenAPI Specをベースに APIを開発
テスト	■ 開発されたAPIのテスト
リリース	■ APIのリリースと変更に応じた対応 ■ リリースに伴う変更管理、利用者などへのコミュニケーション
発見	■ APIカタログなどを通してAPI仕様の公開 ■ API利用者が容易に必要なAPIを特定できる状態の実現
利用	■ APIの死活管理や状況確認が容易にできる状態の実現 ■ API利用中のトラブルへの対応
分析/最適化	■ APIの利用状況を分析できる状態の実現 ■ APIの利用状況に応じたベネフィットやコストの検証

API仕様の原則

標準化に向けた原則

原則	詳細
APIはセキュアである	設計されたAPIはセキュアである必要がある。標準に沿ったAPIは常に誤って他人の情報が漏れないように設計される必要がある。
オープンスタンダードで設計	より多くの利用シーンでAPIが活用される事を実現にあたり RFC, ISOなどコミュニティーで普及している標準を可能な範囲で適用する。
より良いデベロッパーエクシピリエンスを実現	新しいデベロッパーが安易にAPI利用ができる要素を取り入れる。誰もが理解しやすいAPI仕様を提供し、開発がスムーズに行える事をAPI標準によって実現する

API仕様の原則（続き）

標準化に向けた原則（テクニカル）

原則	詳細
REST型のAPI	ステートレスやリソースを中心としたAPI設計をREST型で可能な限り提供
実装は自由	APIの実装方式API仕様によって左右されるべきでは無い。 逆に実装された開発言語やテクノロジーがAPIを通して漏れない事も必要だある。
シンプルである	複雑なAPI程API**の提供側と利用側**に発生するコストと時間がかかる。シンプルである事でコストを最小限に抑えるべきである。
バージョンコントロールされ、下位互換性がある	新しいデータ項目や機能追加の際、既存のAPI利用者に影響がないようにする必要がある。 影響がある場合はバージョンコントロールによって制御され、期限のある下位互換性のポリシーを設ける事で既存の利用者は一定期間守られる。

API開発標準

開発における共通的な仕様

原則	詳細
アーキテクチャスタイル ・通信プロトコル	アーキテクチャスタイルとしてREST、通信プロトコルとしてHTTPsの仕様を推奨。
データ表現形式	JSONを推奨。簡素かつ軽量に構造化したデータを記述可能であり、新たに開発されるAPIにおいてはJSONが主流。
認証・認可プロトコル	OAuth2.0認可フレームワークを推奨する。
バージョンコントロールされ、下位互換性がある	新しいデータ項目や機能追加の際、既存のAPI利用者に影響を生じさせないこと。やむを得ず影響がある場合は下位互換性のポリシーを設けた上で、バージョンコントロールによって制御することを推奨。

API定義

・API仕様を確立するために決定すべき一般的な設計項目は、以下に示すとおりである。
・農機共通API仕様書に記載すべき項目は、ここから取捨選択する必要がある。

No	設計項目		詳細	今年度対応	検討状況
	大項目	中項目			
1	HTTPメソッド、ヘッダー、ステータス	HTTPメソッド	特定のリソースに対して実行する必要があるアクションを示すためのリクエスト・メソッドのセットを定義	○	検討済み
2		HTTPヘッダー(リクエスト)	HTTP のメッセージヘッダーの要求メッセージ項目と値のペア	要検討	ー
3		HTTPヘッダー(レスポンス)	HTTP のメッセージヘッダーの応答メッセージに表示される項目と値のペア	要検討	ー
4		HTTPステータスコード	HTTPステータスコードは3桁の数字で、サーバが返す応答のタイプに基づいて5つのグループに分類されます	○	検討済み
5	バージョン	URIバージョン	APIのURIに関わるバージョンはURI構造で定義されるバージョン	○	検討済み
6		APIバージョン	APIへの変更を透過的に管理するための方針	○	検討済み
7		ドキュメントバージョン	API仕様変更をドキュメントで管理するための方針	要検討	ー
8		HTTPヘッダーバージョン	呼び出し先全体に関わるバージョン	○	検討済み
9	URI	URI構造	APIを呼び出すためのURI構造を決める	○	検討済み
10		クエリパラメター	クエリパラメータは、Webサーバーに情報を伝えるためにURLに追加する情報	○	検討済み

API定義（続き）

API仕様を確立するために決定すべき一般的な設計項目

No	設計項目 大項目	設計項目 中項目	詳細	今年度対応	検討状況
11	ID		特定のリソースのデータ値を表す一意の値	○	検討済み
12	エラー		エラー発生時に返すレスポンスのデータ項目を確定	○	検討済み
13	APIセキュリティ	認証・認可	認証・認可として要求者によってAPIリソースへのアクセス制限できる仕組	○	検討中
14		セキュリティ	APIセキュリティとして暗号化、CORS対応等対応できる仕組み	要検討	–
15	大量データ	ページネーション	大量のレコードを返す各APIエンドポイントの対応方針	要検討	–
16	レスポンス項目		要求されたデータに加えて、応答に追加で返す情報	要検討	–
17	標準データ項目		大文字と小文字、空の値とNULL、日付とタイムスタンプ、列挙（ENUM)	○	検討済み
18	命名規則		API実装の際に使える命名規則が明確である。 例：URI、リソース、クエリ等の命名規則についてルール定義	要検討	–
19	英語対応		API実装に対して英語対応できる状態	要検討	–
20	バイナリデータ	画像, Audio	バイナリファイルは、連続したバイト形式でデータを格納するために使用されるファイルの一種	要検討	–