令和７年度地理空間情報の連携環境の構築に向けたアーキテクチャ調査業務
報告書

令和 8 年 3 月 30 日

株式会社 NTT データ

令和 7 年度地理空間情報の連携環境の構築に向けたアーキテクチャ調査業務

目　次

第1章　はじめに

1.1 業務概要

1.1.1 業務の目的

本調査は、令和 6 年 6 月 21 日の閣議決定に基づき、地理空間情報の利活用を推進するために実施し、「建築・都市の DX」官民ロードマップの中で新たに位置づけられた『不動産 ID を用いた建築・都市分野の多様なデータの連携促進』を支える連携環境の整備を目的としている。

主管課様は、令和 6 年 4 月に不動産ライブラリの運用を開始し、様々な不動産情報を重畳・可視化することによって、地理空間情報の公開を進めている。さらなるデータ利活用を目指し、『産学官全ての主体が地理空間情報を容易に連携し、高度な分析を行うことができる環境を構築』することを目的とした予備調査を実施した。

本調査は、予備調査で判明した課題とユースケースを基に連携環境の要件を抽出し、令和 10 年度の運用開始に向けた初期整備の第一歩とすることを目的とする。

1.1.2 業務概要

(1) 業務名：令和 7 年度地理空間情報の連携環境の構築に向けたアーキテクチャ調査

(2) 工　期：令和 7 年 5 月 23 日から令和 8 年 3 月 31 日まで

(3) 発注者：国土交通省　不動産・建設経済局　地理空間情報課

(4) 受注者：株式会社 NTT データ

1.1.3 業務項目

業務項目は下記の通りである。

1. 業務計画書作成
2. 予備調査において整理された課題の仕分け
3. 不動産 ID への要求仕様の整理および連携キー仕様の検討
4. システム構成検討
5. 机上検証
6. 機能ブロック図の作成
7. 最終報告書作成

1.2 業務実施方針

1.2.1 業務フロー

以下に示すフローにしたがって業務を実施した。なお、「7. 机上検証」は、「3. システム構成検討」および「4. 不動産 ID 導入検討」と並行して実施する。

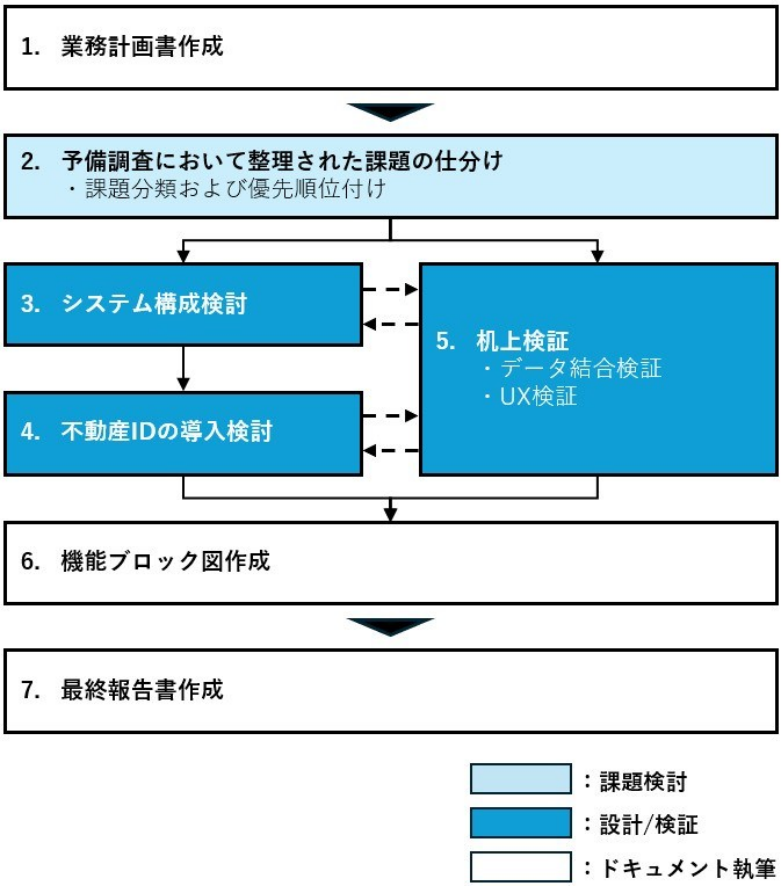

図 1.2-1 業務フロー

1.2.2 業務内容

1）業務計画書作成

本業務を着手するに先立ち、業務の目的・趣旨を把握した上で、業務の目的を達成するために業務の進め方等について十分検討し、業務計画書を作成、提出する。なお、本業務の実施にあたっては「デジタル社会推進標準ガイドライン群」を参照するものとする。

2）予備調査において整理された課題の仕分け

「令和 6 年度地理空間情報データ連携環境の構築に向けた予備調査」において、「概念データモデル」と「実データモデル」を比較し抽出された課題に対し、「連携環境によって対応すべきもの」と「連携環境の整備による解決になじまないもの」に分類する。「連携環境によって対応すべきもの」については、改修ロードマップを作成し、各課題の対応時期を明記する。

3）システム構成検討

連携環境のシステム構成の検討を実施する。(4) で作成するプロトタイプツールの設計書に加え、次年度以降の拡張性に合わせたシステム構成案を検討する。作成する成果物は以下である。

① システム構成図

プロトタイプツール／最終システムの 2 段階をブロック図で表現する。
机上検証結果をもとに、令和 8 年度以降のシステム構築を見据え、ガバメントクラウドで求められる AP のモダン化等に耐えうる構成に検討する。

② 機能一覧

システムの対象となる機能を一覧表に整理する。

③ アクター定義書

機能処理に関与する関係者(アクター)と関与のレベルを整理し、具体化する。

④ 外部インターフェース仕様書

システム構成図、アクター定義書等をもとに、連携環境のシステム化対象範囲の境界部が備えるべき外部インターフェースを整理する。

⑤ データフロー図

各種カタログサイト ⇄ 連携環境 ⇄ ユーザという流れを可視化する。

4）不動産 ID の導入検討

不動産 ID は、現時点で仕様や実現性判断の優先度は定まっていないため、本年度のプロトタイプツール構築および次年度以降の構築では、座標をキーとしてデータ紐づけを実施する想定である。不動産 ID が整備された後、連携環境で各種データを紐づける際に使用する連携キーを、「座標」から「不動産 ID」に変更するための改修が必要になる。本フェーズでは、不動産 ID の最新の整備状況を確認し、連携環境

A) 不動産 ID の整備状況の確認

不動産 ID の検討は現在継続中であり、現時点で最終的な仕様や整備方針は未確定と認識している。不動産市場整備課様に対し、最新の不動産 ID の整備状況を確認し、整理する。

B) 不動産 ID を連携キーとする場合の影響調査

連携キーを「座標」から「不動産 ID」に変更した場合に、連携環境の影響範囲を調査し、改修が必要な箇所を整理する。

C) 不動産 ID への要求仕様の整理

(A)、(B) の内容を踏まえ、連携環境から不動産 ID への要求仕様を整理する。

5）机上検証

連携環境が有効に機能することを確認するとともに、システムの一層の改善を図るため、「データ結合検証」と「UX 検証」を実施する。プロトタイプツールの連携対象データである「不動産情報ライブラリ」「PLATEAU」「登記所備付地図」それぞれの仕様確認およびデータ取得方法の検討を実施する。

1. データ結合検証

「2. UX 検証」で作成するプロトタイプツールの連携対象データである「不動産情報ライブラリ」「PLATEAU」「登記所備付地図」それぞれの仕様およびデータ取得方法を整理する。また、それらのデータを座標によって抽出する必要があるため、それぞれの抽出方法を検討する。なお、仕様書には「成果物として、データ結合検証で使用したデータを提出」との記載があるが、本調査では一般公開されているデータのみを対象としているため、データの提出は実施しない。

2. UX 検証

連携環境のプロトタイプツールを作成する。プロトタイプツールでは、「不動産情報ライブラリ」「PLATEAU」「登記所備付地図」それぞれからデータを取得し、ユーザが指定した座標に紐づくデータを自動抽出し返却する。また、返却されたデータを地図画面上に表示し、座標によって抽出されたデータを可視化する。

6）機能ブロック図作成

当初整備システム及び最終システムにおける全体像を視覚的に整理するために、機能ブロック図を作成する。この機能ブロック図は、(3) で定義したシステム構成図および機能一覧をベースとし、さらに (5) で実施した机上検証の結果を踏まえて構成するものである。

各機能や処理の流れを明確に可視化することで、システム全体の構成関係や役割

分担を把握しやすくし、令和 8 年度以降に予定される仮システム構築において、実装設計の基礎資料として活用できる精度・粒度で整備する。また、プロトタイプツール／最終整備といった整備段階ごとの差異や機能の進化を重ね合わせて表現することで、段階的な発展シナリオを視覚的に共有できるよう工夫する。

7）報告書作成

（1）～（6）の成果を要領よくとりまとめ、報告書（A4 判 2 頁の概要版を含む）を作成するものとする。報告書の作成にあたっては、個人情報や公表しないことを前提に知り得た情報、著作権の取り扱いに留意し、必要な対応をとる。

報告書の作成にあたっては、文書、図、表、その他の業務による記述については、全てワードプロセッサー、図形描画ソフト、表計算ソフト、プレゼンテーションソフト等を用いる。また、既存の資料については可能な限り電子データとして入手する。

なお、報告書のうち、対応する電子ファイルが存在する全ての部分については、対応する電子ファイルを CD-R に格納するものとする。但し、電子データとして入手することが困難な既存の資料については、調査職員と協議する。

第2章　予備調査において整理された課題の仕分け

2.1 課題の仕分け

2.1.1 令和 6 年度予備調査において整理された課題

令和 6 年度に実施された「地理空間情報データ連携環境の構築に向けた予備調査」では、産学官全ての主体が地理空間情報を容易に連携し、高度な分析を行うことができる環境を構築するために必要な要素や、その構築に当たって課題となる事項の洗い出しを実施した。同予備調査では、官民様々な不動産関連の既存データソースを対象に、それぞれのデータ仕様や提供形態を調査するとともに、8 つの想定ユースケースを設定し、各ユースケースにおいて必要となるデータや機能を整理した。その結果、データの収集、加工、連携、活用の各段階において、多岐にわたる課題が抽出された。

抽出された課題は、「別紙 3_データ課題の抽出」において 117 件のデータ課題として、また「別紙 5_最適な連携環境と主要課題」において 50 件のその他の課題として整理されており、合計 167 件の課題が確認されている。

本章では、これらの課題に対し、「連携環境によって対応すべきもの」と「連携環境の整備による解決になじまないもの」に分類する。連携環境の整備による解決になじまない課題とは、データ提供者側での対応が必要な課題や、技術的・制度的な制約により現時点では対応が困難な課題等を指す。その上で、連携環境の整備によって対応すべき課題に対しては、対応方針および対応時期を明記したロードマップを作成し、令和 10 年以降の運用開始に向けた段階的な構築計画を示す。

2.1.2 分類方法

本節では、予備調査において「別紙 3_データ課題の抽出」および「別紙 5_最適な連携環境と主要課題」それぞれに記載された課題に対する分類方法を整理する。

「別紙 3_データ課題の抽出」の「1. データ課題一覧」に記載された課題を「データ課題」、「別紙 5_最適な連携環境と主要課題」に記載された課題を「その他の課題」として、分類方法を以下にまとめる。

1）データ課題の分類方法

データ課題に対して、「データ連携環境で対応するべき課題」と「データ連携環境の対応範囲外」に分類する。

各データ課題の対応主体を整理し、「対応主体」カラムに「データ連携基盤」が含まれる課題は「データ連携環境で対応するべき課題」、含まれない課題を「データ連携環境の対応範囲外」として分類する。

続いて、「データ連携環境で対応するべき課題」それぞれの対応時期を「R7,8 対応」と「R9 以降対応」に分類する。そのために、「別紙 3_データ課題の抽出」の「想定されるデータ」カラムをもとに、各データ課題の対象データ分類を整理する。「対象データ分類」カラムが「オープンデータ（今年度の連携対象）」の課題は「R7,8 対応」、「オープンデータ（今年度の連携対象外）」または「クローズドデータ」の課題は「R9 以降対応」として、「対応時期」カラムに記載する。

2）その他の課題の分類方法

その他の課題については、全て連携環境の対応範囲内であるため、対応時期の整理のみ実施する。各課題の「今年度の構築に対する影響の有無」を整理し、「有」と分類された課題は「R7,8 対応」、「無」と分類された課題は「R9 以降対応」とする。

2.1.3 分類結果

2.1.2 の分類方法に基づき、予備調査で抽出された全 167 件の課題を分類した結果を「別紙_2-1-3_R6 予備調査課題整理」に整理した。

1）データ課題の分類方法

データ課題 117 件について分類を行った結果、以下のように整理された。

① 連携環境で対応すべき課題：45 件

R7,8 年度対応：11 件

R9 年度以降対応：34 件

② 連携環境の対応範囲外：72 件

連携環境の対応範囲外とされた 72 件の課題は、主に以下のような理由により、連携環境の整備では解決が困難なものである。

・データ提供者側での対応が必要な課題（データのオープン化、API 整備等）

・個人情報やプライバシーの取り扱い上、収集が困難な課題

・民間企業が保有するデータで、外部連携が前提ではない課題

・技術的な制約により、現時点では実現が困難な課題

2）その他の課題の分類結果

その他の課題 50 件については、全て連携環境の対応範囲内であるため、対応時期の分類のみを実施した。

① R7,8 年度に対応する課題：17 件

② R9 年度以降に対応する課題：33 件

3）分類結果の総括

全 167 件の課題のうち、連携環境で対応すべき課題は 95 件（データ課題 45 件＋その他の課題 50 件）となった。内訳は以下のとおりである。

この結果から、今後の連携環境の構築においては、まず R7,8 年度に対応すべき 28 件の課題を優先的に取り組み、段階的に R9 年度以降対応の 67 件の課題に対応していくことが適切であると判断される。

分類	R7,8 年度対応	R9 年度以降対応	対応範囲外	合計
データ課題	11 件	34 件	72 件	117 件
その他の課題	17 件	33 件	-	50 件
合計	28 件	67 件	72 件	167 件

表 2.1-1 課題分類結果

2.2 課題の対応方針

分類された各課題に対する対応方針を整理した。特に、R7,8 年度に対応が必要な 28 件の課題については、本調査における対応内容を明確にし、次年度以降の本格運用に向けた具体的な方針を定めた。各課題の方針内容については、「別紙_2-1-3_R6 予備調査課題整理」を参照されたい。

2.2.1 今年度に影響がある課題の対応方法

R7,8 年度対応として分類された 28 件の課題のうち、特に今年度のプロトタイプツール開発および次年度の本格システム構築に直接的な影響を与える課題については、本調査の各章において優先的に検討・実装を行った。

1）今年度に影響がある主要な課題と、本調査における対応状況

今年度に影響がある主要な課題と、本調査における対応状況は以下である。

A） 第 3 章（システム構成検討）にて対応した課題
- ・データ取得方法・データ量・更新頻度の整理
- ・バッチ処理の実装方針の策定
- ・バッチ処理の実装方針の策定

B） 第 4 章（机上検証（データ結合検証））において対応した課題
- ・不動産情報ライブラリ・PLATEAU・登記所備付地図のコードマスタ確認
- ・ジオコーダー・逆ジオコーダーの比較検討と選定
- ・表記揺れへの対応方法の検討
- ・建物ポリゴンとの紐づけ方法の検討

C） 第 5 章（机上検証（UX 検証））において対応した課題
- ・プロトタイプツールの実装と動作検証
- ・データ取得から可視化までの一連のプロセスの検証

2）次年度以降の引継ぎ事項

本調査において検討・実装を行った結果、次年度以降の本格システム構築においては、以下の点に留意する必要がある。

- プロトタイプツールで検証した処理方式を、本格システムにおいても適用可能であることを確認
- バッチ処理の実装においては、処理時間やリソース使用量を考慮した設計が必要
- バッチ処理の実装においては、処理時間やリソース使用量を考慮した設計が必要

2.2.2 改修ロードマップ

分類・整理された 95 件の対応すべき課題に対する改修ロードマップを作成した。ロードマップでは、各課題の対応時期、優先度、関連する他の課題との依存関係を整理し、令和 10 年度の運用開始に向けた段階的な連携環境の構築計画を示している。

1）ロードマップの基本方針

改修ロードマップは、以下の基本方針に基づいて策定した。

① 段階的な機能拡張
　初期整備フェーズで基本機能を実装し、拡張フェーズで高度な機能を追加

② データソースの段階的拡大
　オープンデータから開始し、段階的にクローズドデータへ拡大

③ 技術検証の重視
　新しい技術や手法については、小規模な検証を経てから本格導入

④ ユーザフィードバックの反映
　各フェーズでユーザからのフィードバックを収集し、次フェーズに反映

2）フェーズ別の対応計画

R7,8 年度（初期整備フェーズ）では、連携環境の基本機能を実装し、オープンデータ（今年度連携対象）の連携を実現することを目標とする。主な取り組み内容は以下である。

R7 年度	プロトタイプツールでの実装検証	座標指定によるデータ取得機能の実装
		3 種のデータソース（不動産情報ライブラリ、PLATEAU、登記所備付地図）からのデータ取得
		データ変換・統合処理の実装
		MCP によるデータ抽出機能
		QGIS での可視化機能の検証

R8 年度	本格システムの設計・構築	プロトタイプツールの検証結果を踏まえたシステム設計
		ガバメントクラウドへの対応
		ユーザ管理機能の実装
		運用体制の構築

表 2.2-1 初期整備フェーズの改修計画

R9 年度以降（拡張フェーズ）では、データソースを拡大し、高度な分析機能を追加することを目標とする。主な取り組み内容は以下である。

R9,10 年度	データソースの拡大・機能追加	オープンデータ（今年度連携対象外）の追加
		クローズドデータの連携検討
		不動産 ID によるデータ取得
		QGIS での可視化機能の検証
R11 年度以降	高度な分析機能の追加	時系列データの分析機能
		3D 都市モデルを活用した高度なシミュレーション
		VR・AR を活用した可視化機能

表 2.2-2 拡張フェーズの改修計画

3）ロードマップの実現に向けた留意事項

ロードマップ実現に向けては、以下の点に留意する必要がある。

① 関係機関との調整

データ提供者や関係機関との調整には一定の時間を要するため、早期から協議を開始する。

② 技術動向の注視

AI・クラウド等の技術は急速に進化しているため、常に最新動向を注視し、必要に応じて計画を見直す。

③ ユーザニーズの変化への対応

ユーザニーズは時間とともに変化するため、柔軟に対応する。

第3章　システム構成検討

3.1 成果物

3.1.1 システム構成図

POC、初期構築（プロトタイプツール)、最終システムそれぞれのシステム構成を別紙_3-1-1_システム構成図に示す。検討するに当たって、以下の点を留意した。

1）検討の方針について

データソースより取得するデータは件数が多く、処理をシーケンシャルに実行すると相当な時間がかかる。エンドユーザからのリクエストに対して、返却までのリードタイムを短くしたい。そこで、取得したデータをデータ毎、さらにデータ内の地物毎に分割し、それぞれを別のコンピュートリソースで実行することで並列度を上げる。

2）本システムのインターフェースについて

エンドユーザが指定した緯度経度を元に、以下のデータソースよりデータを取得する。

- 不動産ライブラリ
- PLATEAU
- 登記所備付地図（G 空間情報センター提供の API）

取得したデータは、変換、座標付与（座標が存在しない情報の場合）を行い、返却用のファイルを作成後、エンドユーザにファイルのダウンロード URL をメールにて返却する。

ポイントデータについては、エンドユーザが指定した緯度経度から特定半径内に存在するデータに絞って返却する。この特定半径は、緯度経度と同様にエンドユーザが指定をできる。一方、ポリゴンデータについては、指定した緯度経度と重なるものに絞って返却する。

メールの送信先は、エンドユーザがリクエスト時にリクエスト内に記載する。

3）データソースの API について

① 不動産ライブラリ

不動産ライブラリが提供する API では、XYZ のタイル座標を指定し情報取得が可能。API の返却内容からエンドユーザへ返却する内容に変換する処理コストも低い。そのため、エンドユーザからのリクエストに対して同期的に取得・

変換が可能な状況である。

よって、リクエストを受信した処理内で、取得・変換を行い、配布対象のオブジェクトストレージに格納する。

② PLATEAU

PLATEAU が提供する API では、メッシュコードを指定して検索が可能。しかし、返却内容に記載の URL から、別途、ファイルを取得する必要がある。加えて、ファイル内の情報にある立体ポリゴンから、底面ポリゴンへの変換が必要となり、総じて処理コストが高い。

よって、非同期のバッチ処理にて取得・変換を行う必要がある。変換を行った情報は、短期インデックスに格納する。

③ 登記所備付地図

G 空間情報センターで提供される登記所備付地図の API では、市区町村名を指定して検索を行うことしかできない。対象範囲が広く、返却内容に記載のファイル一覧から、別途、ファイルを取得する必要がある。加えて、ファイル内の情報を GeoJSON へ変換、任意座標の場合、地番からジオコーダーで緯度経度を算出する必要があり、総じて処理コストが高い。よって、非同期のバッチ処理にて取得・変換を行う必要がある。変換を行った情報は、短期インデックスに格納する。

エンドユーザからのリクエスト時、PLATEAU、登記所備付地図については、短期インデックスにデータが存在しなければ、バッチ処理の実行をフックする。バッチ処理にて情報の取得・変換をし、短期インデックスに格納する。

4) バッチ処理について

① API へのリクエスト

データソースの API に過分な負荷をかけないよう、同時リクエスト数を制御する仕組みが必要となる。PLATEAU、登記所備付地図それぞれに対応するキューを用意し、API へのリクエストをキューに登録する。キューから取り出す件数を制御することで、同時リクエスト数を制御する。

API への初回リクエストは、取得対象のファイル一覧が返却される。このファイル毎に API へリクエストする必要がある。初回リクエストの結果、ファイル取得のリクエストもキューへ登録する。

最終的にファイルの内容を取得した後、そのファイルをオブジェクトストレージに格納する。この時点では、無加工のオリジナルのファイルが格納され

る。

② 変換処理

API にて取得したファイルがオブジェクトストレージに格納されたイベントをフックし、変換処理を実行する。変換処理には、1 次変換と 2 次変換が存在する。

- 1 次変換
  オリジナルの内容からフォーマット変換、地物単位への分割を行う。処理結果は、地物単位にオブジェクトストレージに格納される。
- 2 次変換
  1 次変換後のファイルについて、コード値リストの変換、座標が存在しない地物について座標の付与を行う。処理結果は、オブジェクトストレージに格納される。

③ 短期インデックスへの登録

2 次変換後のファイルについて、座標により検索ができるよう短期インデックスへ登録する。

5）短期インデックスについて

PLATEAU、登記所備付地図について、バッチ処理にて変換されたデータが格納される。変換処理にコストがかかるため、変換後のデータを短期的に格納することが主目的となる。

よって、データに対して TTL を設けられる DB が好ましい。また、DB については、マスタ的な管理を必要としないので、厳密な ACID 特性を備えたものである必要はない。リレーショナル DB は ACID 特性を備えており、その分データの登録に処理コストがかかり、同時接続数の制約も厳しい。そこでリレーショナル DB でなく、検索に特化したものが好ましい。

上記のような事情はあるが、プロトタイプツールの実装において、準備が容易であることから、Amazon RDS for PostgreSQL を用いた。

6）情報の統合とエンドユーザへの通知について

エンドユーザに返却するデータが短期インデックスに揃ったタイミングで、短期インデックスより配布対象のファイルを抽出し、先に取得した不動産ライブラリのファイルと共に zip に圧縮、1 ファイルとする。その後、zip ファイルへのリンクが記載されたメールをエンドユーザへ送信する。

7）POC と初期構築（プロトタイプツール）での差異部分について

初期構築では、短期インデックスのストレージとして Amazon RDS for PostgreSQL を用いている。

3.1.2 機能一覧

別紙_3-1-2_機能一覧に、機能の一覧を示す。

3.1.3 アクター定義書

別紙_3-1-3_アクター定義書に、アクターの定義を示す

3.1.4 外部インターフェース定義

別紙_3-1-4_外部インターフェース定義書に、外部インターフェースの定義を示す。リクエストボディの target_apis（地理空間情報取得リスト）に指定するコードは、別紙_3-1-4_地理空間情報取得コードに示す。

3.1.5 データフロー図

別紙_3-1-5_データフロー図に、データフロー図を示す。

3.1.6 その他のシステム設計書

以下の 3 つの設計資料については、本調査では作成していないが、関連する内容の調査を実施し、以下の章にまとめている。

1）**非機能要求整理表**

第 8 章の「8.7.6 セキュリティ要件」にて、インフラレイヤーおよびアプリケーションレイヤーで満たすべきセキュリティ要件を整理している。

2）**連携キーマトリクス表**

「第 7 章 業務効率化効果検証」にて、各ユースケースの業務効率化イメージについて記載している。事業者ごとに「理想のデータ取得方法」も併せて整理している。

3）**品質許容基準表**

「第 7 章 業務効率化効果検証」にて、各ユースケースで取得ニーズが高いデータを整理している。

第4章　机上検証（データ結合検証）

4.1　データ仕様調査

4.1.1 不動産情報ライブラリデータ仕様

不動産ライブラリに関する API 仕様およびデータ構造を把握し、今後のシステム設計やデータ連携に必要な基礎情報を整理することを目的として調査を実施した。調査では、各 API のデータ項目、取得方法、入力条件、出力形式ならびに課題と対応策を確認し、一覧化した。これにより、不動産ライブラリを活用する際の仕様理解を深め、開発における設計判断の参考とすることを目指した。

1）データ一覧

不動産ライブラリにおける各 API のデータ（データの詳細、返却時の座標情報、API を使用するか、返却データの単位）を整理した一覧表を作成した。詳細は別添資料の「別紙_4-1-1_不動産ライブラリデータ一覧」を参照するものとする。

2）不動産ライブラリの課題

検証段階において挙げられた課題に関して、不動産ライブラリの API 毎に課題と対応案をまとめた一覧を作成した。詳細は別添資料の「別紙_4-1-1_不動産ライブラリの課題」を参照するものとする。

3）不動産ライブラリデータ仕様

不動産ライブラリに関する各 API の仕様（パラメータ、内容、入力例、必須項目、出力項目、出力例、備考、連携キーになりうる項目）を整理した。詳細な仕様については、別添資料の「別紙_4-1-1_不動産ライブラリデータ仕様」を参照するものとする。

■調査結果一覧

●：座標情報あり

▲：市区町村コード等、紐づけ可能な情報あり

×：紐づけ不可

API 一覧	結果
1. 不動産価格（取引価格・成約価格）情報取得 API	▲
2. 都道府県内市区町村一覧取得 API	▲
3. 鑑定評価書情報 API	●
4. 不動産価格（取引価格・成約価格）情報のポイント（点）API	●
5. 地価公示・地価調査のポイント（点）API	●
6. 都市計画決定 GIS データ（都市計画区域/区域区分）API	●
7. 都市計画決定 GIS データ（用途地域）API	●
8. 都市計画決定 GIS データ（立地適正化計画）API	●
9. 国土数値情報（小学校区）API	●
10. 国土数値情報（中学校区）API	●
11. 国土数値情報（学校）API	●
12. 国土数値情報（保育園・幼稚園等）API	●
13. 国土数値情報（医療機関）API	●
14. 国土数値情報（福祉施設）API	●
15. 国土数値情報（将来推計人口 250m メッシュ）API	●
16. 都市計画決定 GIS データ（防火・準防火地域）API	●
17. 国土数値情報（駅別乗降客数）API	●
18. 国土数値情報（災害危険区域）API	●
19. 国土数値情報（図書館）API	●
20. 国土数値情報（市区町村役場及び集会施設等）API	●
21. 国土数値情報（自然公園地域）API	●
22. 国土数値情報（大規模盛土造成地マップ）API	●
23. 国土数値情報（地すべり防止地区）API	●
24. 国土数値情報（急傾斜地崩壊危険区域）API	●
25. 都市計画決定 GIS データ（地区計画）API	●
26. 都市計画決定 GIS データ（高度利用地区）API	●
27. 国土交通省都市局（地形区分に基づく液状化の発生傾向図）API	●

表 4.1-1 調査結果一覧

4.1.2 PLATEAU データ仕様

PLATEAU データに関する API 仕様およびデータ構造を把握し、今後のシステム設計やデータ連携に必要な基礎情報を整理することを目的として調査を実施した。調査では、各 API のデータ項目、取得方法、入力条件、出力形式、ならびに課題と対応策を確認し、一覧化した。これにより、PLATEAU データを活用する際の仕様理解を深め、開発における設計判断の参考とすることを目指した。

1）データ一覧

PLATEAU における API のデータ（データの単位、取得方法、検索条件、入力値、返却データの単位、抽出範囲、課題）を整理した一覧表を作成した。詳細は「別紙_4-1-2_PLATEAU データ一覧」を参照するものとする。

2）データ仕様

PLATEAU に関する API の仕様（ファイル構成、地物ごとのデータ項目、連携キーになりうる項目等）を整理した。詳細な仕様については「別紙_4-1-2_PLATEAU データ仕様」および「別紙_4-1-2_PLATEAU ファイル構成」を参照するものとする。

3）属性情報調査

PLATEAU データが持つ「LOD レベル」への理解を深めるための調査を実施し、地物・LOD レベルごとの属性情報の構成を整理した。また、座標情報取得時における LOD レベルの取得優先度についても検討を行い、処理方針の明確化を図った。詳細な仕様については「別紙_4-1-2_PLATEAU 属性情報」を参照するものとする。

■LOD レベルについて（地物：bldg（建築物）の場合）

LOD レベル	内容	提供エリア
LOD0	平面に投影したもの。高さ情報がない。	すべてのエリア
LOD1	最も単純な直方体の組み合わせで構成されたモデル。	すべてのエリア
LOD2	外観の凸凹が表現され、屋根や壁などを再現したモデル	特定のエリア
LOD3	ドアや窓などの開口部、道路の立体交差なども表現。	限定されたエリア
LOD4	建物の外観だけでなく、内部もモデル化。	一部の建築物

表 4.1-2LOD レベルについて

4.1.3 登記所備付地図データ仕様

登記所備付地図データに関する API 仕様およびデータ構造を把握し、今後のシステム設計やデータ連携に必要な基礎情報を整理することを目的として調査を実施した。調査では、各 API のデータ項目、取得方法、入力条件、出力形式、ならびに課題と対応策を確認し、一覧化した。これにより、登記所備付地図データを活用する際の仕様理解を深め、開発における設計判断の参考とすることを目指した。

1）データ一覧

登記所備付地図における API のデータ（データの単位、取得方法、検索条件、入力値、返却データの単位、抽出範囲、課題）を整理した一覧表を作成した。詳細は「別紙_4-1-3_登記所備付地図データ一覧」を参照するものとする。

2）データ仕様

登記所備付地図に関する API の仕様（ファイル構成、データ項目、連携キーになりうる項目等）を整理した。詳細な仕様については「別紙_4-1-3_登記所備付地図データ仕様」および「別紙_4-1-3_登記所備付地図ファイル構成」を参照するものとする。

4.2　データ取得調査

4.2.1 不動産情報ライブラリデータ取得方式

不動産ライブラリのデータ取得方法について、任意の 1 地点を設定し、実際にどのようなデータが取得可能かを検証したものである。取得方法の確認、実装手順、処理時間、取得結果の内容について整理をした。

1）調査対象のデータ取得結果

以下の 1 地点に対し、不動産ライブラリのデータを取得した。取得したデータやパラメータ設定時に生じた問題について後続で調査を実施している。データ取得結果の詳細は「別紙_4-2-1_調査対象のデータ取得結果」を参照するものとする。

■対象情報

住所	〒156-0043 東京都世田谷区松原 2 丁目 9-11 フォレスト松原		
緯度経度	35.66425470840267, 139.64709341949828		
XYZ タイル	29094,12905,15		
メッシュコード	53393591		
市区町村コード	13112	丁目コード	023002

表 4.2-1 対象情報

2）リクエストごとのデータ取得方法

不動産ライブラリのデータ取得においては検索条件の指定方法として以下 3 パターンが存在する。緯度経度、XYZ タイル、メッシュコードを検索条件とした際のデータ取得方法をまとめている。ユーザからの入力値は緯度経度のリストであることを前提とし、今後のデータ取得処理は「緯度経度」による検索を基本方針として進める。

■データ取得方法

検索条件	不動産ライブラリ(取得方法)
緯度経度 例）35.657961，139.605305	緯度経度 → XYZ タイル座標に変換し、XYZ タイル座標で API 検索
XYZ タイル 例）X=29091，Y=12905，Z=15	XYZ タイル座標で API 検索
メッシュコード 例）53393488（3 次メッシュ）	メッシュコード → 緯度経度(メッシュの中心) → XYZ タイル座標 に変換し、XYZ タイル座標で API 検索

表 4.2-2 データ取得方法

3）**リクエストが XYZ タイル以外の API について**

本件は API パラメータの設定に関して検討をした。API パラメータを大きく分類すると別添資料の「別紙_4-2-1_XYZ タイル以外のパラメータ調査」の【API パラメータ】の通りとなっている。その中で、XYZ タイル以外のパラメータを設定する API について調査を実施した。

① 不動産価格（取引価格・成約価格）情報取得 API

入力パラメータの検討および、不動産価格情報に関しては不動産価格（取引価格・成約価格）情報取得 API と不動産価格（取引価格・成約価格）情報のポイント（点）API が存在していたため、レスポンス内容を比較し差異があるか調査を実施した。

- 入力パラメータの検討

  必須パラメータは取引時期（年）＋ 都道府県・市区町村・駅コードのいずれかとなっている。駅コードと市区町村コードの 2 パターンで検討を実施している。調査の詳細は別添資料の「別紙_4-2-1_XYZ タイル以外のパラメータ調査」を参照するものとする。

- レスポンス内容の比較
  不動産価格（取引価格・成約価格）情報取得 API と不動産価格（取引価格・成約価格）情報のポイント（点）API のレスポンス内容は取得できる内容が異なっていた。詳細は別添資料の「別紙_4-2-1_不動産価格情報レスポンス内容比較」を参照するものとする。

上記の調査結果および別添資料「別紙_4-2-1_調査対象のデータ取得結果」の不動産価格（取引価格・成約価格）情報のポイント（点）API の結果から、不動産価格（取引価格・成約価格）情報のポイント（点）API は駅ごとにデータを保持しており、駅の座標で検索しないと結果が得られないことが判明した。そのため、不動産価格（取引価格・成約価格）情報取得 API を使用し、市区町村コードを入力パラメータとして設定することで広範囲のデータを取得してから住所で絞り込む方針とする。

② 都道府県内市区町村一覧取得 API
都道府県内市区町村一覧取得 API の必須パラメータは都道府県コードとなっている。パラメータで指定する都道府県コードの取得方法を以下 2 パターンで検討した。プロトタイプツールでは取得方法 2 の逆ジオコーディングでの取得を採用する方針とする。

- 取得方法 1：行政区域ポリゴンとの照合
  緯度経度を、国土数値情報（行政区域データ）が保有するポリゴンデータと重ねて都道府県を判定する。判定には PostGIS 関数 ST_Covers(a, b) を使用し、行政区域ポリゴン（a）が対象地点または対象ポリゴン（b）を完全に覆っているかを確認する。ST_Covers(a, b) は、b のすべての点が a の内部または境界上にある場合に TRUE を返す。詳細は別添資料の「別紙_4-2-1_XYZ タイル以外のパラメータ調査」の 2. 都道府県内市区町村一覧取得 API を参照するものとする。

- 取得方法 2：逆ジオコーディングによる取得
  対象地点の緯度経度を逆ジオコーディングし、市区町村コードを取得する。取得した市区町村コードの先頭 2 桁を抽出することで、対応する都道府県コードを取得できる。抽出した都道府県コードを入力パラメータとして設定し、都道府県内市区町村一覧取得 API にリクエストを実施する。

③ 鑑定評価書情報 API

鑑定評価書情報 API は必須パラメータが都道府県コード、価格時点（年）、用途区分となっている。都道府県コードに関しては、②都道府県内市区町村一覧取得 API 同様に逆ジオコーディングをして市区町村コードを取得し、そこから抽出した都道府県コードをパラメータとして設定する。

4）複数タイルに跨るポリゴンデータについて

不動産ライブラリのポリゴン情報を返却する API について、返却されるポリゴン情報の範囲を確認するため調査を実施した。調査の結果、対象ポリゴンが複数の XYZ タイルに跨る場合、API はポリゴン全体ではなく、検索したタイル内に含まれる部分のみを返却することが判明した。詳細の調査内容は別添資料の「別紙_4-2-1_複数タイルに跨るポリゴンデータ」を参照するものとする。

5）地形区分に基づく液状化の発生傾向図 API の調査

不動産ライブラリの国土交通省都市局（地形区分に基づく液状化の発生傾向図）API に対し、地図表示におけるズームレベル 14 と 15 の挙動差異について調査を実施した。ズームレベル 14 で取得可能な液状化傾向データが、ズームレベル 15 では検索したタイル内に存在していても取得できないケースがあるとのことで、両ズームレベル間でのデータ取得の網羅性に差異があるかを検証した。調査方法としては、10 拠点を対象にズームレベル 14 と 15 で API を実行し、取得結果を比較している。結果として、差異は確認できなかった。詳細の調査内容は別添資料の「別紙_4-2-1_地形区分に基づく液状化の発生傾向図 API の調査」を参照するものとする。

6）周辺タイルを検索した場合

検索座標がタイルの境界付近に位置する場合、検索対象タイル内の情報のみでは、実際に近接する地物を見落とす可能性がある。特に、検索タイル内に該当データが存在しない場合でも、隣接するタイルに有効な情報が含まれているケースがあることから、検索精度の向上を目的として、周辺タイルを含めた検索の有効性について調査を実施した。

- 調査方法
  - ① 検索座標が属する XYZ タイルに加え、以下 2 パターンでデータを取得
    - A） 近隣する 8 タイルを含めた計 9 タイル
    - B） 近接する 3 タイルを含めた計 4 タイル
  - ② 各パターンで検索結果を比較・検証を実施。

詳細な調査手法および結果については、別添資料「別紙_4-2-1_周辺タイル検索」と「別紙_4-2-1_不動産ライブラリのデータ取得結果」を参照するものとする。

- 調査方法
  検索座標が境界付近にある場合において、隣接タイルに有効な地物情報が存在するケースが複数確認された。これにより、検索対象を拡張することで、より正確かつ網羅的な検索結果が得られることが明らかとなった。

- 対応方針
  以上の結果を踏まえ、半径が約 500m 以内の情報が必要な場合は 4 タイル分の情報で有効的な情報が取得できるため、検索座標が属する XYZ タイルに加え、近接する 3 枚のタイルを含めた 4 タイルを対象とした検索方式を採用する。今後さらに広い範囲を抽出する必要がある場合は、9 タイル分の情報を取得する方式や、ズームレベルを下げて広域をカバーする方式などの検討が必要となる。

7）各 API のデータ取得結果の整理

前述の調査でデータ取得方針が定まったことから、各 API に対してリクエストを実行し、計測時間および取得結果を一覧でまとめた。

- 対象条件
  基本的には下記対象情報の地点を対象としているが、この地点では結果が得られない API に関しては別地点で検索を実施している。

- 記載内容
  ・ポリゴン情報のタイプがポイント、ラインで返却される API
  →周辺タイルを含めた検索を実施し、半径 500m 以内の地物件数を記載している。
  ・ポリゴン情報のタイプがポリゴンで返却される API
  →検索座標と重なる地物の件数を記載している。
  ・ポリゴン情報がない API
  →全体の取得結果から住所で抽出した件数を記載している。
  詳細は別添資料の「別紙_4-2-1_不動産ライブラリのデータ取得結果」を参照するものとする。

■対象情報

住所	〒157-0071 東京都世田谷区千歳台 6 丁目 11-16 パークハウス世田谷コンフォート		
緯度経度	35.657961, 139.605305		
XYZ タイル座標	29091, 12905, 15		
メッシュコード	53393488		
市区町村コード	13112	丁目コード	046006

表 4.2-3 対象情報

4.2.2 PLATEAUデータ取得方式

PLATEAUデータの取得方法について、任意の1地点を設定し、実際にどのような地理空間情報が取得可能かを検証したものである。取得方法の確認、実装手順、処理時間、取得結果の内容について整理した。

1）データ取得結果（手動検証）

以下の 1 地点に対し、該当する市区町村の PLATEAU データを手動でダウンロードした。取得したデータをもとに、検索座標と空間的に重なる建築物情報を抽出した。この検証により、PLATEAUデータから検索座標に関連する建築物情報が取得可能であることを確認できた。

取得データの詳細は「別紙_4-2-2_PLATEAUデータ取得結果（手動検証）」を参照するものとする。

住所	東京都世田谷区千歳台 6 丁目 11-16 パークハウス世田谷コンフォート
緯度経度	35.657961, 139.605305
XYZ タイル座標	29091/12905/15
メッシュコード	53393488
市区町村コード	13112

表 4.2-4 対象情報

2）リクエストごとのデータ取得方法

PLATEAUのデータ取得においては、ユーザが指定する検索条件に応じて、対象データの抽出方法が異なる。主に以下3つの検索パターンが存在し、それぞれに対応した変換処理を行うことで、データ取得が可能となる。

■検索パターン

① 緯度経度
　地図上の任意の地点を指定する方法

② XYZ タイル座標
　タイル状に分割された地図の 1 タイルを指定する方法

③ メッシュコード
　格子状に区分されたエリアの 1 区画を指定する方法

それぞれの検索条件に対して、どのように変換処理を行い、該当するデータを取

得するかを整理した。

■データ取得方法

検索条件	データ取得先
	PLATEAU
緯度経度 例）35.657961, 139.605305	【取得方法】 緯度経度とポリゴンが重なるデータを取得 【考慮事項】 データ量が多いため、ある程度絞り込みが必要。 （メッシュコード等）
XYZ タイル 例）X=29091, Y=12905, Z=15	【取得方法】 XYZ タイル座標をポリゴンに変換し、 ポリゴン（変換後）とポリゴン（PLATEAU データ）が重なるデータを取得 【考慮事項】 データ量が多いため、ある程度絞り込みが必要。 （メッシュコード等）
メッシュコード 例）53393488 （3 次メッシュ）	【取得方法】 メッシュコードでデータを取得 【考慮事項】 検索条件が何次メッシュかにより変換が必要 （Python で変換可能）

表 4.2-5 データ取得方法

ユーザからの入力値は緯度経度のリストであることを前提とし、今後のデータ取得処理は「緯度経度」による検索を基本方針として進める。

なお、PLATEAU データは地物数が多く、ファイルサイズも大きくなる傾向があるため、処理負荷や取得時間、データの抽出・変換方法などについても、課題として検討を進める必要がある。

3）データ取得処理の実装手順と課題

ユーザが入力した緯度・経度（検索座標）を基に、PLATEAU データを取得し、結果を返却するまでの実装フローを整理した。

■実装フロー

1. 緯度・経度を XYZ タイル（ズームレベル 18）に変換
2. PLATEAU API で対象データを検索し、該当ファイルをダウンロード
3. 各地物から LOD1 レベルの座標情報を抽出
4. 抽出した座標情報から底面ポリゴンを抽出
5. 抽出した底面ポリゴンと検索座標の重なりを判定
6. 属性情報のコード値をコードリストに基づき変換
7. 判定結果をファイルとして出力

   ※PLATEAU API についての調査内容は「別紙_4-2-2_PLATEAU_API 調査」を参照するものとする。

また、実装フローの整理に伴い、以下の課題についても検討した。

■課題

1. 重なり判定の座標情報について

   PLATEAU データの座標情報は緯度・経度・高さから構成される 3 次元の立体座標である。そのため、検索座標との重なりを判定するには、判定可能な形式への変換が必要となる。判定方法としては以下の 2 つが考えられる。

   - 方法①

     立体座標から地物の中心点を抽出し、検索座標との一致や距離（〇m 以内）で判定する方法。

   - 方法②

     地物の底面となるポリゴン情報を抽出し、検索座標との空間的な重なりを判定する方法。

   方法①は点と点による判定となるため、精度や柔軟性に限界がある。一方、方法②は点と面による判定が可能であり、より正確な空間判定が行えるため、本実装では底面ポリゴンを用いた重なり判定方式を採用する。詳細は「4.3.2 PLATEAU データ抽出方式」に記述する。

2. コードリストの変換方法について

   PLATEAU データの属性情報には、用途地域や建物種別などを示すコード値

が含まれている。これらをユーザが理解可能な値へ変換するには、PLATEAUが提供するコードリストとの照合が必要となる。

しかし、コードリストは市区町村ごとに内容が異なるため、マスタ情報として DB 上に保持するのは現実的ではない。本実装では変換処理の都度、対象のコードリストファイルを参照し、該当するコード値を照合・変換する方式を採用する。

3. 座標情報抽出の処理時間について

各地物の LOD1 レベルの座標情報を抽出し、底面ポリゴンとの重なり判定を行う一連の処理において、特に地物数が多いエリアでは処理が長くなる傾向がある。PLATEAU データの構造が複雑であることやファイルサイズが大きいことが主な要因として挙げられる。

このパフォーマンス課題に対して、以下のような改善策を検討する必要がある。

・3DCityDB などの 3D 都市モデル管理ツールの活用

CityGML 形式を GeoJSON に変換し、座標抽出や空間判定処理の効率化を図る。

・PLATEAU API への機能追加要望

検索座標に重なる地物情報を直接返却する仕組みを実現してもらう。

4）データ取得処理の実行時間計測

3)で整理したデータ取得処理の実装手順に基づき、試作版を用いて処理時間の計測を行った。検索地点は、初期設定した任意の地点「東京都世田谷区千歳台 6 丁目 11-16 パークハウス世田谷コンフォート」とする。

■計測結果

No.	処理内容	測定時間
1	API ダウンロード ・緯度経度を変換（XYZ タイル） ・XYZ タイルで API 検索、ファイルダウンロード	318.81 秒 （約 5 分）
2	CityGML 解析 ・各地物の LOD1 の座標情報抽出 ・底面ポリゴン抽出 ・緯度経度との重なり判定	7192.12 秒 （約 2 時間）
3	出力（その他処理） ・コードリスト値の値変換 ・ファイル出力	0.03 秒
	合計	**7,505.96 秒 （約 2 時間 5 分）**

表 4.2-6 計測結果

なお、3)で課題として挙げた通り、CityGML の解析処理に最も多くの時間を要しており、全体処理時間の大部分を占めていることが確認できた。今後は、処理の効率化に向けた手法（例：3DCityDB の活用やデータ形式の変換）について、さらなる検討が必要である。

5）処理結果のデータ概要

データ取得処理の実行結果としては、検索座標と重なった地物データが合計 6 件取得できた。内訳は以下の通りである。

・bldg（建築物）：1 件
・luse（土地利用）：2 件
・urf（都市計画決定情報）：3 件

4.2.3 登記所備付地図データ取得方式

登記所備付地図データの取得方法について、任意の 1 地点を設定し、実際にどのような地理空間情報が取得可能かを検証したものである。取得方法の確認、実装手順、処理時間、取得結果の内容について整理した。

1）データ取得結果（手動検証）

以下の 1 地点に対し、該当する市区町村の登記所備付地図データを手動でダウンロードした。取得したデータをもとに、検索地点と空間的に重なる建築物情報を抽出した。この検証により、登記所備付地図データから地点に関連する建築物情報を取得可能であることが確認できた。

詳細は別途資料の「登記所備付地図データ取得結果（手動検証)」を参照するものとする。

住所	東京都世田谷区千歳台 6 丁目 11-16 パークハウス世田谷コンフォート
緯度経度	35.657961, 139.605305
XYZ タイル座標	29091/12905/15
メッシュコード	53393488
市区町村コード	13112

表 4.2-7 対象情報

2）リクエストごとのデータ取得方法

登記所備付地図のデータ取得においては、ユーザが指定する検索条件に応じて、対象データの抽出方法が異なる。主に以下 3 つの検索パターンが存在し、それぞれに対応した変換処理を行うことで、データ取得が可能となる。

■検索パターン

① 緯度経度
地図上の任意の地点を指定する方法

② XYZ タイル座標
タイル状に分割された地図の 1 タイルを指定する方法

③ メッシュコード
格子状に区分されたエリアの 1 区画を指定する方法

それぞれの検索条件に対して、どのように変換処理を行い、該当するデータを取得するかを整理した。

■データ取得方法

検索条件	データ取得先 登記所備付地図
緯度経度 例）35.657961, 139.605305	【取得方法】 緯度経度とポリゴンが重なるデータを取得 【考慮事項】 データ量が多いため、ある程度絞り込みが必要。 （住所、市区町村コード、大字丁目コード等）
XYZ タイル 例）X=29091, Y=12905, Z=15	【取得方法】 XYZ タイル座標をポリゴンに変換し、 ポリゴン（変換）とポリゴン（地図データ）が重なるデータを取得 【考慮事項】 データ量が多いため、ある程度絞り込みが必要。 （住所、市区町村コード、大字丁目コード等）
メッシュコード 例）53393488 （3 次メッシュ）	【取得方法】 メッシュコードをポリゴンに変換し、 ポリゴン（変換）とポリゴン（地図データ）が重なるデータを取得 【考慮事項】 データ量が多いため、ある程度絞り込みが必要。 （住所、市区町村コード、大字丁目コード等）

表 4.2-8 データ取得結果

ユーザからの入力値は緯度経度のリストであることを前提とし、今後のデータ取得処理は「緯度経度」による検索を基本方針として進める。
なお、登記所備付地図データは市区町村単位で、ファイルサイズも大きくなる傾向があるため、処理負荷や取得時間、データの抽出・変換方法などについても、課題として検討を進める必要がある。

3）データ取得処理の実装手順と課題

ユーザが入力した緯度･経度(検索座標)を基に、登記所備付地図データを取得し、結果を返却するまでの実装フローを整理した。

■実装フロー

1. 緯度・経度を市区町村名に変換
2. API で該当ファイルを検索し、ダウンロード
3. コンバータでダウンロードデータを GeoJSON 形式に変換
4. 変換後のデータを DB へ登録
5. 緯度・経度と重なるデータを抽出
6. 抽出結果をファイルとして出力

また、実装フローの整理に伴い、以下の課題についても検討した。

■課題

1. 緯度・経度から市区町村名への変換方法について

   登記所備付地図の API を実行する際には、パラメータとして市区町村名の指定が必要となる。これを緯度・経度から取得する方法として次の 2 つが考えられる。

   - 方法①

     逆ジオコーダーを利用して、緯度・経度から住所情報を取得し、その中から市区町村名を抽出する方法。

   - 方法②

     逆ジオコーダーを利用して、市区町村コードを取得し、不動産ライブラリが提供する「都道府県内市区町村一覧取得 API」を用いて、該当する市区町村名を取得する方法。

   方法①は住所表記に揺れが生じる可能性があるため、より安定した取得が可能な方法②を採用する。

2. データの絞り込みについて

   登記所備付地図のデータは市区町村単位で構成されているため、重なり判定を行う前に、ある程度の絞り込みを行わないと処理時間が長くなる傾向がある。

   データ内には住所に関する情報（市区町村コード・大字・小字・地番）が含まれており、これらの単位で絞り込みが可能かどうかを調査した結果、以下の 2 つの方法が考えられる。

・ 方法①
逆ジオコーダーを利用して、緯度・経度から住所情報を取得し、登記所備付地図のデータと照合する方法。
・ 方法②
国土数値情報の「行政区域データ」等のポリゴン情報を使用して、緯度・経度と重なる地域の住所を取得し、登記所備付地図のデータと照合する方法。

方法②では行政区域のポリゴンデータを事前に保有・管理するという運用上の負担があるため、今回は方法①を採用する。

4）データ取得処理の実行時間計測

3）で整理したデータ取得処理の実装手順に基づき、試作版を用いて処理時間の計測を行った。
検索地点は、初期設定した任意の地点「東京都世田谷区千歳台 6 丁目 11-16 パークハウス世田谷コンフォート」とする。

■計測結果

No.	処理内容	測定時間
1	API ダウンロード ・緯度経度変換（市区町村） ・市区町村で API 検索、ファイルダウンロード	115.54 秒 （約 1 分 55 秒）
2	GeoJSON 変換 ・コンバータで地図 XML→GeoJSON 変換	210.40 秒 （約 3 分 30 秒）
3	DB 登録 ・GeoJSON を DB に登録	18930.96 秒 （約 5 時間 15 分）
4	重なり判定 ・緯度経度とポリゴンの重なりを判定	31551.6 秒 （約 8 時間 45 分）
5	ファイル出力 ・該当データを DB から取得し、出力	0.10 秒
	合計	**50,808.6 秒** **（約 14 時間 06 分）**

表 4.2-9 計測結果

なお、今回の重なり判定処理では、3）で課題として挙げていた「データの絞り込み」を実施していないため、処理時間が大幅に増加する結果となった。プロトタイプツール作成においては、この点を踏まえた絞り込み処理の導入により、処理効率の改善を図る予定である。

5）処理結果のデータ概要

データ取得処理の実行結果としては、検索座標と重なった登記所備付地図データが１件取得できた。

4.3　データ抽出・変換方式検討

4.3.1 不動産情報ライブラリデータ抽出方式

不動産ライブラリ API から返却される各種データに対して実施する抽出および変換処理のロジックを整理したものである。座標情報の有無、ジオメトリタイプ（ポリゴン・ポイント・ライン）、および返却形式（GeoJSON／JSON）を考慮し、QGIS 上での可視化を前提としたデータ抽出および形式変換の一連の技術的プロセスについて記述している。各 API の座標情報の有無とジオメトリタイプに関しては「別紙_4-1-1_不動産ライブラリデータ一覧」の座標情報項目を参照するものとする。

1）座標情報を含むデータ

不動産ライブラリ API の返却結果には、座標情報を含むものと含まないものが存在する。ここでは座標情報を含むデータについて整理を行う。座標情報を持つデータはポリゴン、ポイント、ラインの 3 種類のジオメトリタイプで返却されるため、それぞれに応じた抽出方法を適用する。

- ポリゴン情報が返却される場合
  ポリゴン情報を含むデータに対しては、検索座標と返却されたポリゴンとの重なりを判定し、重なりが確認できるデータのみを抽出する。重なりの判定は、検索座標（ポイント）が返却されたポリゴンの底面に含まれているかどうかを基準として行う。

- ポイントおよびライン情報が返却される場合
  ポイントおよびライン情報を含むデータに対しては、検索座標（ポイント）との距離を算出し、指定された半径距離以内に位置しているデータのみを抽出する。
  ポイントの場合は検索座標との直線距離を算出し、ラインの場合は検索座標からラインを構成する線分までの最短距離（垂直距離）を算出して判定を行い、その距離が指定された半径以内であれば抽出する。

2）座標情報を含まないデータ

座標情報を含まないデータに対しては、検索座標（ポイント）を用いて逆ジオコーディングを実施し、取得した住所情報をもとに該当するデータを抽出する。QGIS 上での可視化を前提としているため、抽出したデータには返却されたデータの住所情

報をもとに座標情報を付与する必要がある。

- 不動産価格（取引価格・成約価格）情報取得 API

  この API は座標情報を含まず、JSON 形式でデータが返却されるため以下の手順で処理を行う。

  手順①：検索座標を逆ジオコーディングし、大字までの住所を取得する。

  手順②：不動産価格（取引価格・成約価格）情報取得 API にて返却された JSON 形式のデータから、Prefecture、Municipality、DistrictName の各項目の値を抽出する。

  手順③：抽出した住所と検索座標の住所を照合し、一致するデータのみを抽出する。

  手順④：抽出した住所に対してジオコーディングを行い、代表点の緯度経度を取得する。

  手順⑤：各データを Feature としてまとめ、geometry の"coordinates"に取得した緯度経度、"type"に"Point"を設定し、JSON 形式から GeoJSON 形式に変換する。

  上記手順で使用するジオコーダーと逆ジオコーダーについては、後述の「登記所備付地図データ抽出方式」にて調査内容を示す。

3）JSON 形式で返却されるが座標情報を含むデータ

鑑定評価書情報 API で返却されるデータは座標情報を含んでいるが、返却形式が JSON 形式であるため、GeoJSON 形式への変換が必要となる。以下の手順で処理を行う。

- 鑑定評価書情報 API

  手順①：返却結果の各データを Feature としてまとめ、geometry の"coordinates"に「位置座標 経度」「位置座標 緯度」の値を設定し、"type"に"Point"を指定し、JSON 形式から GeoJSON 形式へ変換する。

  手順②：変換後、検索座標と付与された座標との距離を算出し、指定された半径以内に位置しているデータのみを抽出する（処理内容は、座標情報を持つポイント・ラインと同様）。

4.3.2 PLATEAUデータ抽出方式

PLATEAUが保有する座標情報に対する抽出・変換ロジックの詳細を検証したものである。検索座標との重ね合わせ方法、CityGMLデータの形式変換、コードリスト値の変換処理など、データ抽出に至る一連の技術的プロセスを整理した。

1）検索座標との重ね方について

PLATEAUデータから地物の底面ポリゴンを抽出し、検索座標との空間的な重なりを判定する処理方式について詳述する。

- 抽出対象のポリゴン情報

  PLATEAUデータはCityGML形式で提供されており、各地物は複数のポリゴンで構成される（3次元座標）。これらのポリゴンのうち、底面に該当するポリゴンを抽出することで、2次元平面上での空間判定が可能となる。

- 底面ポリゴンの抽出手順

  以下の手順で底面ポリゴンを抽出する

  手順①：ポリゴン頂点の抽出

  各地物に含まれるポリゴンの頂点座標を取得

  手順②：最低高度ポリゴンの特定

  抽出したポリゴンのうち、Z座標（高さ）が最も低いポリゴンを底面とみなす。建築物の場合、床の基礎部分に相当するため、空間判定に適している。

  手順③：2次元ポリゴンへの変換

  底面ポリゴンの頂点座標からZ要素（高さ）を除去し、2次元の平面ポリゴンとして扱う

- 検索座標との重なり判定

  変換された底面ポリゴンと検索座標（点）との空間的な関係を判定する。具体的には以下のような処理を行う

  ・検索座標が底面ポリゴンの内部に含まれているかどうかを判定

2）CityGML データの形式変換について

データ返却形式の検討にあたり、CityGML を GeoJSON または CityJSON に変換する方法について調査を行った。また、変換を行うことで座標情報の抽出処理にかかる時間の短縮も期待できる。

- 調査内容

  返却形式の検討にあたり、以下ツールを用いて調査を行った。詳細は「別紙_4-3-2_PLATEAU データ形式変換」を参照するものとする。

  ・FME Form

  地理空間データの構造や内容変換・統合が可能なソフトウェア。

  ・ogr2ogr

  地理空間データのベクタ形式や座標系の変換が可能なコマンドラインユーティリティ。

  ・3DCityDB

  3D 都市モデルをデータベースで管理・分析するためのオープンソースパッケージ。

  ・plateaukit

  3D 都市モデルデータを扱うための Python ライブラリ。

  ・citygml-tools

  CityGML 形式の 3D 都市モデルデータを処理するためのコマンドラインユーティリティ。

- 調査結果

  最終的に、QGIS 上での表示を前提とした場合、3D 情報を保持できる形式が望ましいと判断されたため、CityGML 形式のまま返却する方針とした。

3）コードリスト値の変換処理について

PLATEAU データの属性情報に含まれる、用途地域や建物種別などを示すコード値の変換処理について詳述する。

- コードリストファイルについて

  コードリストの定義は、/codelists フォルダに xml 形式で配置されている。各コードリストファイルには、属性値に対応するコードとその説明が定義されている。

- 変換処理について

  PLATEAU データの属性には、codeSpace にコードリストファイル名が記載されており、属性値にはコード値が定義されている。処理時には、指定されているコードリストファイルを参照し、コード値を人が理解可能な値へ変換する。

  例）

  PLATEAU データ（CityGML）

```
<bldg:class
codeSpace="../../codelists/Building_class.xml">3001</b
ldg:class>
```

  コードリスト（XML）

```
<gml:Definition gml:id="Building_class_1">
<gml:description>普通建物</gml:description>
<gml:name>3001</gml:name>
</gml:Definition>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
```

  上記の例では、コード値「3001」は「普通建物」として変換処理を行う。

- 変換結果の比較

  コードリスト値変換を行った場合と行わない場合について、QGIS 上で属性情報を参照した際の表示内容を比較した。PLATEAU のプラグインを利用してインポートしたデータでは、変換の有無による表示結果に差は見られなかった。一方、通常のレイヤとしてインポートしたデータでは、変換を行った場合に変換後の値が適切に表示される結果となった。利用時には不動産ライブラリや登記所備付地図など他データと併せて通常レイヤとしてインポートする運用を想定しているため、コードリスト値変換処理を行う方針とする。詳細については「別紙_4-3-2_PLATEAU コードリスト値の変換結果比較」を参照するものとする。

4.3.3 登記所備付地図データ抽出方式

登記所備付地図データが保有する座標情報に対する抽出・変換ロジックの詳細を検証したものである。検索座標との重ね合わせ方、地図 XML のデータ形式変換、ジオコーダー・逆ジオコーダーなど、データ抽出に至るまでの一連の技術的プロセスを整理した。

1）地図 XML データ形式変換について

登記所備付地図データは、地図 XML 形式で提供されている。この形式は構造が非常に複雑であり、座標情報をプログラム上で直接取得するのが困難である。そのため、デジタル庁が提供している変換コンバータを利用し、GeoJSON 形式に変換する処理を組み込む方針とした。このコンバータは、地図 XML からアドレス・ベース・レジストリに必要な属性情報のみを抽出して出力する仕様となっている。境界点や境界線などの情報は出力されない。また、座標値は経度・緯度（JGD2011）に変換されるが、任意座標系のデータは座標値として変換されない。

2）座標系について

登記所備付地図データには、公共座標系と任意座標系の 2 種類が存在する。
それぞれの特徴は以下の通りである。

- 公共座標系
  公共座標は、国が定める基準に基づき、地球上での絶対的な位置を示す座標系である。
- 任意座標系
  任意座標は、任意の点を基準とした座標系であり、基準点は測量者が任意に設定できる。そのため、測量成果ごとに基準が異なる特徴を持つ。

3）検索座標との重ね方について

- 公共座標系
  コンバータで変換した GeoJSON データには、ポリゴン情報が含まれている。そのポリゴン情報と、検索座標（点）との空間的な重なりを判定し、重なりがあるデータを返却対象とする。
- 任意座標系
  任意座標は公共座標の基準と異なり、コンバータでは座標値に変換されない。検索座標との重なりを判定するためには、任意座標系のデータを加工する必

要がある。本実装では、任意座標系のデータに含まれる市区町村・大字・小字・地番等の情報をジオコーダーで座標に変換し、検索座標との重なりを判定する方針としている。検討内容の詳細については「別紙_8-2_登記所備付地図の任意座標系」に記述する。

4）ジオコーダー、逆ジオコーダーの利用について

登記所備付地図データの検索や検索座標との重なり判定には、座標と住所情報の相互変換が不可欠である。以下の 2 つの用途で、ジオコーダーおよび逆ジオコーダーを活用する。

- ジオコーダーの利用
  任意座標系のデータに対して、住所から座標（点）を取得し、検索座標との距離を比較することで、対象地物の判定に活用する。
- 逆ジオコーダーの利用
  検索座標から住所情報（市区町村名）を取得し、登記所備付地図データの検索キーとして利用する。

このように、座標から住所、住所から座標の双方向変換を通じて、地理空間データの検索精度と利便性を向上させることを目的としている。

5）ジオコーダー、逆ジオコーダーの比較検討について

住所と座標の相互変換は検索精度の向上に不可欠であり、ジオコーダーおよび逆ジオコーダーの選定はシステムの信頼性・効率性に直結する。
そのため、複数のサービスを対象に、精度・対応範囲・利用制限・コストなどの観点から比較検討を行い、最適な API の選定を試みた。今回の検討では以下の 3 つのサービス（ABR、国土地理院、東京大学提供の jageocoder）を対象とした。

■検証結果の比較表

項目	ABR ジオコーダー	国土地理院	東大（jageocoder）
利用形態	CLI / API	API	CLI / API
ジオコーダー	○	○	○
逆ジオコーダー	×	○	○
精度レベル	都道府県～地番	市区町村～地番	市区町村～地番
データ更新	必要	不要	必要
出力形式	JSON	JSON	JSON / GeoJSON

表 4.3-1 検証結果の比較表

まず、利用形態については、いずれのサービスも API 利用が可能であるが、ABR および東京大学（jageocoder）のサービスは、自前で API を構築する必要がある。そのため、これらのサービスを利用する場合は、API サーバの構築・運用およびデータ更新の対応が必要となる。

次に、逆ジオコーダーの対応状況については、ABR は非対応であり、逆ジオコーディングが必要な用途には適さない。一方、国土地理院および東京大学（jageocoder）は逆ジオコーダーに対応しており、検索座標から住所情報（市区町村名）を取得する用途に適している。また、精度レベルについては、ABR は「都道府県～地番」までと幅広く、国土地理院と東京大学は「市区町村～地番」まで対応している。

実際に任意の地点においてジオコーディングおよび逆ジオコーディングを行った比較調査の詳細な結果については「別紙_4-3-3_ジオコーダーの検証結果」および「別紙_4-3-3_逆ジオコーダーの検証結果」を参照するものとする。

4.4　データ出力方式の検討

4.4.1 出力方式の比較検討

「不動産ライブラリ」、「PLATEAU」、「登記所備付地図」で取得したデータの出力方式について、QGIS などの可視化ツールでの表示を前提に以下の 3 つの返却形式を対象として比較検討を行った。なお、本調査では、各返却形式の比較検討にあたり、QGIS を用いて実際の検証を行っている。

① GeoJSON ファイルにまとめて返却
すべてのデータを GeoJSON 形式にし、GeoJSON 結合を行ったうえで、1 つの GeoJSON ファイルとして返却する方式。
② GeoPackage にまとめて返却
各データをそのまま GeoPackage ファイルにまとめて返却する方式。
③ 各ファイルをそのまま返却
各データファイルを ZIP 形式でまとめて返却する方式。

詳細は「別紙_4-4-1_出力パターン一覧」および「別紙_4-4-1_GeoJSON 結合」を参照するものとする。

検証の結果、①は形状（ポイント、ライン、ポリゴン）ごとにレイヤ分けがされている。しかし、XYZ タイル範囲を超えるポリゴンが含まれる場合、表示上ではそのポリゴンが全面に描画され、他のポリゴンが隠れてしまうことがある。このため、詳細な区別が困難となることが判明した。

一方、②および③はデータごとにレイヤが分かれており、表示順や色分けの設定が可能で視覚的識別が容易である。ただし、②は PLATEAU の 3D 表示に対応できない可能性がある。

以上を踏まえ、③の「各ファイルをそのまま返却する方式」を採用する。

第5章　机上検証（UX 検証）

5.1　UX 検証の概要

本調査では、不動産情報ライブラリ API、PLATEAU、登記所備付地図といった複数の不動産・地理空間データを対象に、指定した座標点に紐づく情報のみを取得・返却するプロトタイプツールを開発した。

これらのデータは、不動産分野や都市計画、エリア分析等において利活用が期待されている一方で、データ提供形態や取得方法がデータソースごとに異なり、ユーザが個別に仕様を理解し実装する必要があるという課題がある。

本検証では、こうした課題に対し、位置情報（座標）を共通の入力インターフェースとすることで、複数のデータソースから関連情報を横断的に取得できる仕組みの実現可能性を検証することを目的とした。

加えて、不動産情報ライブラリ API については、Model Context Protocol（MCP）を用いたサーバを構築し、LLM と連携することで、自然言語によるデータ取得がどの程度有効であるかを検証した。本調査は、将来的な対話型データ利活用の可能性を評価するための技術検証として位置づけている。

5.2　プロトタイプツールの仕様

プロトタイプツールは、ユーザが「座標」と「取得したい情報」を選択すると、その座標点に紐づく情報のみを自動連携するツールである。対象データソースは「PLATEAU」、「登記所備付地図」、「不動産情報ライブラリ」の 3 つである。

詳細な操作手順については、「別紙_5-2_プロトタイプツール操作マニュアル」を参照されたい。

5.2.1 データ抽出方式

データソースごとに、データの特性に応じた抽出方式を採用している。第 4 章で詳細に検討した抽出方式を、プロトタイプツールに実装した。

1）PLATEAU

PLATEAU データは CityGML 形式で提供されており、3 次元の立体座標を持つ。そのため、入力した座標点と重なる底面ポリゴンの抽出を行う。手順は以下のとおりであ

る。

① 緯度・経度を XYZ タイル座標に変換
ユーザが入力した緯度・経度を、ズームレベル 18 の XYZ タイル座標に変換
② PLATEAU API でデータを検索・ダウンロード
XYZ タイル座標をもとに、該当する CityGML ファイルをダウンロード
③ 底面ポリゴンの抽出
各地物の LOD1 レベルの座標情報から、Z 座標（高さ）が最も低いポリゴンを底面として抽出
④ 重なり判定
検索座標（点）が底面ポリゴン（面）の内部に含まれているかを判定
⑤

なお、建築物（bldg）のほかに、土地利用（luse）や都市計画決定情報（urf）等も取得対象とする。

2）**登記所備付地図**

登記所備付地図データは地図 XML 形式で提供されており、座標系が公共座標系と任意座標系の 2 種類存在する。入力した座標点と重なるポリゴンを抽出するため、以下の手順でデータ抽出を行う。

① 緯度・経度を市区町村名に変換
逆ジオコーディングにより、検索座標から市区町村コードを取得し、市区町村名を特定
② API で該当ファイルを検索・ダウンロード
市区町村名をパラメータとし、該当する地図 XML ファイルをダウンロード
③ GeoJSON 形式に変換
デジタル庁提供の変換コンバータを用いて、XML を GeoJSON に変換
④ データベースへ登録
変換後の GeoJSON データをデータベースに登録
⑤ 重なり判定
検索座標と重なるポリゴンを抽出

なお、任意座標系のデータは座標情報が含まれないため、住所情報を用いて間接的に判定（詳細は第 4 章を参照）する。

3）**不動産情報ライブラリ**

不動産情報ライブラリは 30 種類の API を提供しており、各 API のデータ形状タイプ（ポリゴン、ライン、ポイント）に応じて、以下の抽出方式を採用する。

【ポリゴンデータ】

① 緯度・経度を XYZ タイル座標に変換
検索座標を、ズームレベル 15 の XYZ タイル座標に変換

② 周辺タイルを含めて検索
検索座標を含むタイルに加え、近接する 3 タイルを含めた計 4 タイルを検索対象とする

③ API でデータを取得
各タイルに対して API を実行し、GeoJSON 形式でデータを取得

④ 重なり判定
検索座標がポリゴン内に含まれるデータのみを抽出

【ポイントデータ・ラインデータ】

① 周辺タイルを含めて検索
ポリゴンデータと同様に 4 タイル分のデータを取得

② 距離計算
検索座標から各ライン・ポイントまでの距離を計算

③ 範囲内のデータを抽出
ユーザが指定した半径（最大 425m）以内に位置するデータのみを抽出

5.2.2 データ返却形式

取得したデータは、QGIS などの GIS ツールで表示しやすいよう、以下の形式で返却する。

・不動産情報ライブラリおよび登記所備付地図：GeoJSON 形式
・PLATEAU：CityGML 形式（3 次元情報を保持するため）

各データファイルは、ZIP 形式でまとめて返却される。この方式により、ユーザは取得したデータを即座に QGIS 等で可視化・分析することが可能となる。

5.2.3 コンポーネント構成

プロトタイプツールは、以下のコンポーネントで構成される。

1	入力インターフェース	座標入力（緯度・経度）
		取得データ選択
		抽出半径指定(不動産情報ライブラリのポイント・ラインデータ用)
2	データ取得･処理エンジン	各データソースの API 連携
		座標変換処理（XYZ タイル、メッシュコード等）
		空間判定処理（重なり判定、距離計算）
3	出力インターフェース	データのファイル変換・統合
		ZIP 形式での一括返却

表 5.2-1 コンポーネント構成

5.3 MCP サーバの仕様

自然言語によるデータ取得を実現するため、Model Context Protocol（MCP）サーバを構築した。本 MCP サーバは、不動産情報ライブラリ API を連携対象とした。

1）実装機能

MCP サーバには、プロトタイプツールで実装した距離による絞り込み機能を実装している。これにより、自然言語でのポイントデータ取得においても、「○○メートル以内の情報を取得」といった指定が可能となる。また、「API で取得したデータ（GeoJSON 形式）をユーザが指定したフォルダに保存する機能」や、「不動産情報ライブラリの地図画面（指定されたエリア）の URL を自動発行する機能」も実装した。これにより LLM 画面から不動産情報ライブラリの地図画面に遷移することができる。

2）対応 API

今年度は、不動産情報ライブラリで提供される以下の API を連携対象とし、自然言語によるデータ取得を可能にするローカル MCP サーバを構築した。

- 不動産価格（取引価格・成約価格）情報取得 API
- 鑑定評価書情報 API
- 地価公示・地価調査のポイント（点）API
- 都市計画決定 GIS データ（都市計画区域・区域区分）API
- 都市計画決定 GIS データ（用途地域）API
- 都市計画決定 GIS データ（立地適正化計画）API
- 国土数値情報（小学校区）API
- 国土数値情報（中学校区）API

・国土数値情報（学校）API
・国土数値情報（保育園・幼稚園等）API
・国土数値情報（医療機関）API
・国土数値情報（福祉施設）API
・国土数値情報（将来推計人口 250m メッシュ）API
・都市計画決定 GIS データ（防火・準防火地域）API
・国土数値情報（駅別乗降客数）API
・国土数値情報（災害危険区域）API
・国土数値情報（図書館）API
・国土数値情報（市区町村役場及び集会施設等）API
・国土数値情報（自然公園地域）API
・国土数値情報（大規模盛土造成地マップ）API
・国土数値情報（地すべり防止地区）API
・国土数値情報（急傾斜地崩壊危険区域）API
・都市計画決定 GIS データ（地区計画）API
・都市計画決定 GIS データ（高度利用地区）API
・国土交通省都市局（地形区分に基づく液状化の発生傾向図）API
・国土数値情報（洪水浸水想定区域（想定最大規模））API
・国土数値情報（高潮浸水想定区域）API
・国土数値情報（津波浸水想定）API
・国土数値情報（土砂災害警戒区域）API
・国土数値情報（人口集中地区）API

5.4　MCP サーバの動作環境・構築手順

作成した MCP サーバ（milt-geospatial-mcp）の動作環境及び利用手順を以下に示す。

5.4.1 動作環境

本 MCP サーバは、ローカル環境で動作するように設計されている。これにより、以下のメリットがある。

・セキュリティの確保
　データ取得時の座標や住所等の情報がローカルに留まる

・プライバシーの保護
ユーザの問い合わせ内容が外部に送信されない
・高速な応答
ネットワーク遅延が最小限に抑えられる

5.4.2 技術構成

本 MCP サーバは、以下の技術構成で構成されている。

プログラミング言語	Python
MCP プロトコル	Model Context Protocol 仕様に準拠
LLM 連携	Claude Code 等の MCP 対応 LLM ツールと連携
API クライアント	不動産情報ライブラリ API への HTTP リクエスト機能
ジオコーディング	国土地理院等のジオコーディングサービスとの連携

表 5.4-1 技術構成

Python を採用した理由は、距離計算等の位置情報に関する関数が整備されていることが挙げられる。

5.4.3 セキュリティ考慮事項

ローカル環境での動作に加え、以下のセキュリティ対策を実施している。
・API キーの管理
不動産情報ライブラリの API キー（取得が必要な場合）を環境変数で管理
・入力値の検証
不正な座標や異常な範囲指定等をチェック
・エラーハンドリング
異常な状況でも適切にエラー処理を実施

5.4.4 MCP サーバの構築手順

本 MCP サーバの詳細な構築手順については、不動産情報ライブラリ MCP の README.md に記載している。構築手順には、以下の内容が含まれる。

1	環境準備	必要なライブラリのインストール
		開発環境の設定
2	サーバのセットアップ	MCP サーバのインストール
		設定ファイルの編集
3	LLM との連携設定	Claude Code 等の LLM ツールとの接続設定

		認証情報の設定
4	動作確認	サンプルクエリでの動作テスト
		エラーハンドリングの確認

表 5.4-2 構築手順概要

5.5 モックアップ画面イメージ検証結果と考察

本 UX 検証における検証結果を以下にまとめる。

5.5.1 プロトタイプツールの検証結果

プロトタイプツールを用いた検証の結果、以下が確認された。

・座標を共通キーとして、複数のデータソースから関連情報を統合的に取得できることが実証された

・データ取得から可視化までの一連のプロセスが、従来に比べて大幅に簡素化された

・QGIS での可視化により、取得したデータの空間的な関係性を直感的に理解できることが確認された

5.5.2 MCP サーバの検証結果

MCP サーバを用いた自然言語インターフェースの検証では、以下が確認された。

・自然言語での指示により、適切な API が選択され、データ取得が実行されることが確認された

・距離指定等のパラメータも、自然言語から適切に解釈・設定されることが確認された

・対話型のデータ取得により、ユーザの試行錯誤的なデータ探索が容易になることが示唆された

5.5.3 今後の課題

今後の展開に向けて、以下の課題が確認された。

1	処理性能の向上	特に PLATEAU の CityGML データ処理において、処理時間の短縮が必要
2	対象データの拡大	クローズドデータへの対応

		新たなオープンデータソースの追加
3	MCP サーバの 機能拡張と課題	より複雑なクエリへの対応
		ジオコーダー機能の搭載（住所や建物名・駅名から座標への変換は、現状は LLM 側の自動処理に任せているが、精度が良くないことがあるため、ジオコーダー機能を搭載する必要がある）
		より複雑なクエリへの対応
		登記所備付地図への対応

表 5.5-1 今後の課題

第6章　機能ブロック図の作成

6.1 機能ブロック図

別紙_6-1_機能ブロック図に各機能のブロック図を示す。機能ブロックは、システム構成のプロトタイプ版をベースに記載している。

第7章　業務効率化効果検証

7.1　業務効率化の概要

本章では、プロトタイプツールによってどのような業務シーンを効率化することが可能か、各不動産関連業務の有識者にヒアリングを実施した結果を整理する。

7.1.1 ヒアリングの目的

本ヒアリングでは、以下の 2 つの観点から効率化の可能性を評価した。

1. 現時点のプロトタイプツールで効率化できそうな箇所
   現在実装されている機能（座標指定による複数データソースからの情報取得、半径指定による範囲検索等）を用いて、どのような業務プロセスが効率化できるか

2. 今後の機能拡張や連携データ増加によって将来的に効率化できうる箇所
   クローズドデータの連携、高度な分析機能の追加、不動産 ID の導入等により、将来的にどのような業務効率化が期待できるか

7.1.2 ヒアリングの実施概要

ヒアリングは、令和 7 年 10 月～12 月にかけて、9 事業者（有識者）へ実施した。実施内容は以下である。

① プロトタイプツールのデモンストレーション
- 座標指定によるデータ取得機能の実演
- QGIS での可視化デモ
- MCP サーバによる自然言語でのデータ取得デモ

② 現状業務におけるデータ取得・活用の課題ヒアリング
- 業務フェーズごとのデータ取得状況
- 取得データの種類・取得単位・取得範囲
- データ取得における課題

③ プロトタイプツールによる効率化可能性の議論
- 現時点のツールで効率化できる業務の特定
- 期待される効率化効果の定量的・定性的評価

④ 将来的な機能拡張に対する要望ヒアリング

・追加で連携してほしいデータソース

・実装してほしい機能

・不動産 ID への期待

7.1.3 ヒアリング結果の整理方法

ヒアリング結果は以下の観点で整理した。

業務フェーズ	どの業務段階でデータが必要とされるか
取得データ	現状どのようなデータを取得しているか
取得方法	データの取得方法と課題
効率化ポイント	プロトタイプツールで効率化できる業務
将来への期待	機能拡張により効率化が期待される業務
自動取得への期待	自動取得できると嬉しいオープンデータ

表 7.1-1 ヒアリング観点

7.2 想定ユースケース

今回ヒアリングを実施した業種・業務は以下のとおりである。

1	宅地系不動産業	宅地系不動産業務の有識者
2	デベロッパー業	デベロッパー業務の有識者
3	不動産情報サービス	賃貸物件マップサービス事業者 不動産投資サポートサービス事業者 賃貸物件検索サポートシステム開発事業者
4	保険業	損害保険会社
5	小売業	コンビニエンスストア新規店舗出店業務の有識者 ショッピングモール新規店舗出店業務の有識者

表 7.2-1 ヒアリング先の業界

これらの業種は、不動産や地理空間情報を業務上頻繁に活用しており、連携環境の恩恵を受けやすいと想定される業種である。

7.3　各ユースケースにおける効率化イメージ

各業種・業務におけるヒアリング結果から、効率化が期待される業務シーンを整理した。

7.3.1 宅地系不動産業

1）現行業務における課題

宅地系不動産業の有識者へのヒアリングにより、以下の業務フェーズにおける課題が確認された。

【現地調査フェーズ】

・物件の形状、土地の境界、周辺施設等の情報を、様々な媒体から収集する必要があり、時間がかかる。
・現地に行かないと確認できない情報が多い。

【役所調査フェーズ】

・都市計画区域、都市計画道路、ハザード情報、登記情報等を取得するため、役所に訪問してデータを取得する必要がある。
・役所の開庁時間内に訪問する必要があり、業務効率が低下する。

【取引事例・市場調査フェーズ】

・過去の取引価格や地価公示等の情報を、複数のデータベースから個別に検索・取得する必要がある。
・取得したデータの統合・可視化に専門知識が必要。

2）プロトタイプツールによる効率化

ヒアリング結果から、以下の業務において効率化が可能であることが確認された。

A)　現時点で効率化できる業務

【現地調査の効率化】

・PLATEAU を用いて物件の形状を事前に確認可能
・登記所備付地図により土地の境界を事前に確認可能
・周辺施設情報（不動産情報ライブラリの POI 情報）を事前に把握可能
・現地調査の回数や時間を削減

【役所調査の効率化】

・都市計画区域の確認を、オンラインで即座に実施可能

・対象物件にかかるハザード情報（洪水、土砂災害、液状化等）を一括収集

・役所への訪問回数を削減

【取引事例・市場調査の効率化】

・過去の取引価格（不動産取引価格情報）を、座標指定で即座に取得

・地価公示情報を、周辺エリアも含めて一括取得

・重要事項説明資料の作成時間を大幅に短縮

B) 将来的に効率化が期待される業務

【市場動向分析】

・より広範囲のデータ連携（人口統計、経済指標等）により、エリア全体の市場動向を把握

・長期的な価格トレンドの分析が可能に

【リスク評価の高度化】

・詳細なハザード情報との連携により、物件のリスク評価の精度向上

・過去の災害履歴データとの連携により、より正確なリスク評価が可能に

【AI 活用による自動査定】

・取得したデータを AI が分析し、自動的に査定額を算出

・査定業務の大幅な効率化と標準化

3) ヒアリングで挙がった具体的な要望

【登記所備付地図（任意座標系）の対応】

任意座標系のデータも含めて取得できるようにしてほしい

【リアルタイム性】

データの更新頻度を高め、常に最新の情報が取得できるようにしてほしい

4) 理想のデータ取得単位

・座標単位

・住所単位

・土地単位

・部屋単位

7.3.2 デベロッパー業

1）現行業務における課題

デベロッパー業では、開発用地の選定や開発計画の策定において、以下のような課題が存在する。

・用地候補地の法規制調査に時間がかかる。（都市計画、建築基準法等）
・周辺環境の詳細な把握が困難。（3D 都市モデル等）
・複数の開発候補地の比較検討に時間がかかる。

2）プロトタイプツールによる効率化

A） 現時点で効率化できる業務

【用地調査業務】

候補地の座標指定により、都市計画情報、用途地域、建築制限等を一括取得

【周辺環境分析】

PLATEAU の 3D 都市モデルにより、周辺建物の高さや配置を立体的に把握

B） 将来的に効率化が期待される業務

【日照・眺望シミュレーション】

3D 都市モデルと組み合わせた高度な分析

【開発リスク評価】

インフラ情報、災害リスク情報等との統合的な評価

3）理想のデータ取得単位

・住所単位
・土地単位
・任意の範囲でまとめて取得

7.3.3 不動産情報サービス業

1）現行業務における課題

不動産情報サービス業（賃貸物件マップサービス事業者、不動産投資サポートサービス事業者、賃貸物件検索サポートシステム開発事業者）では、以下のような課題が存在する。

・物件情報に紐づく周辺環境データの収集に時間がかかる
・複数のデータソースから情報を取得・統合する必要がある
・データの更新頻度や正確性の維持が困難
・ユーザに提供する付加価値情報の拡充が課題

2）プロトタイプツールによる効率化

A） 現時点で効率化できる業務

【物件周辺情報の自動取得】

物件の座標指定により、周辺施設、ハザード情報、都市計画情報等を一括取得

【物件詳細ページの情報充実】

取得したデータをユーザ向けサービスに自動反映

【データ更新業務の効率化】

複数データソースからの情報取得を自動化

B） 将来的に効率化が期待される業務

【物件評価の高度化】

より多様なデータとの連携により、物件の多面的な評価が可能に

【レコメンド機能の精度向上】

周辺環境データを活用した物件推薦の精緻化

【市場分析サービスの提供】

広域データの活用による市場動向分析サービスの展開

3）理想のデータ取得単位

・部屋単位
・駅単位
・学区単位（駅が少ない地方エリアが対象）

7.3.4 保険業

1）現行業務における課題

損害保険会社では､火災保険等の引受審査において､以下のような課題が存在する。

・建物の災害リスク評価に必要な情報収集に時間がかかる
・複数建物が同一敷地内に存在する場合の個別評価が困難
・申請された住所を座標に変換する必要がある
・異なる地域に存在する複数建物を評価する場合（商業施設等)、各地域の
　ハザード情報をそれぞれ確認するのに手間がかかる

損害保険会社では、申請者からの情報の正確性を確認する業務が手間となっているため、不動産 ID の整備によって建物の履歴を取得することにより、効率化対象の拡大が見込まれる。

2）プロトタイプツールによる効率化

A) 現時点で効率化できる業務

【引受審査業務】

対象建物の座標指定により、ハザード情報（洪水、土砂災害、液状化等）を一括取得

【保険料算定】

取得したリスク情報をもとに、より正確な保険料算定が可能

B) 将来的に効率化が期待される業務

【建物個別評価】

不動産 ID の導入により、同一敷地内の複数建物を個別に評価可能に

【リスクマップ作成】

広域のデータ連携により、エリア全体のリスクマップを作成

【過去データとの比較】

過去と現在の PLATEAU データや衛星画像を比較することにより､被害規模の確認業務を短縮することが可能

3）理想のデータ取得単位

・建物単位
・地番単位
・住所単位

7.3.5 小売業

1）現行業務における課題

小売業（大手コンビニエンスストア、ショッピングモール）の新規店舗出店業務の有識者へのヒアリングにより、以下の業務フェーズにおける課題が確認された。

【建物選定フェーズ（某コンビニエンスストア)】

価格情報、人口 GIS、用途地域、周辺施設、登記所備付地図等、様々な媒体からデータを収集するため、時間がかかる。（リアルタイムのデータが必要）

【建物立体調査フェーズ】

景観、看板の視認性、段差等、現地でなければ確認できない情報が多い

【生活導線確認フェーズ】

店前の人流/交通量、導線の合流点、生活行動等、現地でなければ確認できない情報が多い

【土地選定フェーズ（某ショッピングモール)】

工場跡地情報、空地情報等、各自治体の公開情報をチェックする必要があり、時間がかかる

【周辺調査フェーズ】

学校の位置、ハザード情報、周辺人口、道路渋滞情報、インターチェンジの位置等、様々な媒体からデータを収集するため、時間がかかる

【法令確認フェーズ】

都市計画、用途地域、駐車場の附置義務等、様々な媒体からデータを収集するため、時間がかかる

コンビニエンスストアのように多数の店舗を展開する小売業においては、候補地調査の効率化による効果が大きい。年間数十～数百件の出店を検討する中で、1 件当たりの調査時間が短縮されることで、全体として大きな効率化効果が期待できる。

2）プロトタイプツールによる効率化

A） 現時点で効率化できる業務

【建物選定の効率化（某コンビニエンスストア)】

・登記所備付地図データを座標指定で即座に取得
・周辺施設情報（コンビニ、学校等）を一括取得
・用途地域を即座に確認
・データ収集時間を大幅に短縮
【周辺調査の効率化（某ショッピングモール)】
・学校の位置及び学校区を即座に確認
・ハザード情報（洪水、土砂災害、液状化等）を一括取得
・周辺施設情報を網羅的に把握
【法令確認の効率化】
・用途地域を即座に確認
・都市計画区域を即座に確認
・法令確認に要する時間を短縮

B) 将来的に効率化が期待される業務
【商圏分析の高度化】
・人口動態データ（国勢調査等）との連携により、詳細な商圏分析が可能に
・自社カード会員データと公開データを組み合わせた高度な分析
【交通量分析】
・時間帯ごとの交通量データとの連携
・イベントごとの交通量変動の把握
・店前の人流予測の精度向上
【売上シミュレーションの高度化】
・複合的なデータ分析（人口、交通量、競合店舗、周辺施設等）による売上予測
【自治体情報の自動収集】
・各自治体が公開する空地情報を自動的に収集
・工場跡地、大規模施設の閉鎖情報等の自動収集

3）理想のデータ取得単位
・土地単位
・建物単位

7.4　ヒアリング結果まとめ

本節では、全５分野（宅地系不動産業、デベロッパー業、不動産情報サービス業、保険業、小売業）へのヒアリング結果を横断的に分析し、共通課題と将来展望を整理する。

7.4.1 データ取得単位・方法の比較

各分野で必要とされるデータ取得の単位・方法を整理すると、以下のような特徴が見られた。

業界	主なデータ取得単位	取得範囲	頻度	リアルタイム性の重要度
宅地系不動産	ポイント（座標）	物件周辺	案件ごと	中
デベロッパー	ポイント・エリア	開発候補地周辺	プロジェクトごと	低～中
不動産情報サービス	ポイント（建物・部屋）	物件周辺	継続的	高
保険	ポイント（建物・土地）	建物・土地	契約ごと	中
小売	ポイント・エリア	商圏	出店検討ごと	高

表 7.4-1 データ取得単位・方法

共通点として、すべての業界でポイントによる指定のニーズが高い一方で、エリア指定のニーズも高いことが分かった。エリアについては、駅周辺や学区単位での取得だけでなく、ユーザがデータ取得範囲を任意で指定したいという声もあった。そのため、地図上でユーザが任意の範囲を描画することによる指定方法も候補の一つとして考えられる。

7.4.2 共通する課題の整理

複数分野に共通する課題を以下にまとめる。

業界	データの加工・抽出の手間	データのリアルタイム性確認の手間	現地調査の手間	役所調査の手間
宅地系不動産	○	○	○	○
デベロッパー	○	-	-	○
不動産情報サービス	-	○	-	○
保険	○	-	-	○
小売	○	○	○	-

表 7.4-2 共通する課題

不動産情報サービス業は、データ加工・抽出は社内システムで自動化が進んでいる一方で、データのリアルタイム性の確認や、現状の PLATEAU データからは確認できない情報（前面道路の距離や段差等）を現地で確認する手間が挙げられた。

7.5　機能拡張による効率化範囲の拡大イメージ

本節では、令和 8 年度以降の機能拡張によって実現されうる効率化範囲の拡大イメージを示す。

7.5.1 短期的な機能拡張イメージ（R8～9 年度）

R8～9 年度の短期的な機能拡張として、以下が期待される。

1	対象データの拡大	他官公庁及び自治体データとの連携
		ユーザの社内システムとの連携
2	紐づけの高度化	登記所備付地図（任意座標系）の紐づけ率改善
3	データ取得方法の拡張	複数地点の一括検索
		範囲による複数データ取得
4	AI/LLM の活用	自然言語 IF によるデータ取得
		自動評価・予測機能

表 7.5-1 短期的な機能拡張イメージ

AI/LLM の活用については、今年度構築した MCP サーバの機能拡張が含まれる。また、データ取得方法の拡張では、

7.5.2 中長期的な機能拡張イメージ（R10 年度以降）

R10 年度以降の中長期的な機能拡張として、以下が期待される。

1	対象データの拡大	民間保有データとの連携
		業界固有データとの連携
2	不動産 ID の導入	建物単位での履歴情報の追跡
		不動産 ID を連携キーとしたデータ取得
3	リアルタイムデータとの連携	リアルタイム更新情報との連携
		IoT センサーデータ

表 7.5-2 中長期的な機能拡張イメージ

第8章　令和 8 年度以降の構築に向けた検討

8.1 不動産情報ライブラリのデータ取得範囲の拡張

現在の仕様では、抽出範囲（半径距離）の上限値は半径 425m に設定されており、検索座標を含むズームレベル 15 の 4 タイルから情報を取得している。ズームレベルは固定値であり、任意に指定することはできない。このため、取得できる情報は半径 425m 以内に限定されている。

- 課題

  ズームレベルを指定できない現状では、半径 425m より広範囲の情報を取得することができない。さらに広域の情報を取得するためには、ズームレベルを変更できる仕組みが必要である。しかし、ズームレベルを変更するだけでは、抽出範囲が固定されている限り有効性が乏しく、ズームレベルと抽出範囲を連動させる設計が不可欠である。

- 対応案

  広範囲の情報取得を可能にするため、以下 2 つの方針を提案する。

  ① ズームレベルをリクエストパラメータで指定可能にする。
  任意のズームレベルを指定できるようにする。

  ② 半径距離に応じてズームレベルを自動調整する。
  各ズームレベルで対応可能な半径距離を調査し、距離に応じてズームレベルを決定する。

- 考慮点

  対応案を検討するにあたり、以下の点に留意する必要がある。各 API のズームレベルは下記表を参照するものとする。

  ・ズームレベルを下げると広範囲の情報は取得できるが、極端に下げると不要なデータが増え、処理効率が低下する。
  ・ズームレベルを変更するだけでは意味がなく、抽出範囲（半径距離）との対応が必要。
  ・不動産ライブラリの API はそれぞれ指定可能なズームレベルの最小値が異なる。
  ・ズームレベル 13～15 はすべての API で対応可能なため、最小値を 13 に統一する案もあるが、抽出半径距離は全 API で一定のため、API ごとに最小値を変えるメリットは限定的。

■指定可能ズームレベル一覧

API	ズームレベル
地価公示・地価調査のポイント（点）	13～15
都市計画決定 GIS データ（都市計画区域/区域区分）	11～15
都市計画決定 GIS データ（用途地域）	11～15
都市計画決定 GIS データ（立地適正化計画）	11～15
国土数値情報（小学校区）	11～15
国土数値情報（中学校区）	11～15
国土数値情報（学校）	13～15
国土数値情報（保育園・幼稚園等）	13～15
国土数値情報（医療機関）	13～15
国土数値情報（福祉施設）	13～15
国土数値情報（将来推計人口 250m メッシュ）	11～15
都市計画決定 GIS データ（防火・準防火地域）	11～15
国土数値情報（駅別乗降客数）	11～15
国土数値情報（災害危険区域）	11～15
国土数値情報（図書館）	13～15
国土数値情報（市区町村役場及び集会施設等）	13～15
国土数値情報（自然公園地域）	9～15
国土数値情報（大規模盛土造成地マップ）	11～15
国土数値情報（地すべり防止地区）	11～15
国土数値情報（急傾斜地崩壊危険区域）	11～15
都市計画決定 GIS データ（地区計画）	11～15
都市計画決定 GIS データ（高度利用地区）	11～15
国土交通省都市局（地形区分に基づく液状化の発生傾向図）	11～15

表 8.1-1 指定可能ズームレベル一覧

8.2 登記所備付地図データの抽出パターン

8.2.1 距離と住所の抽出比較

登記所備付地図の任意座標系データは公共座標系とは異なり、座標情報を持たないため、検索座標との重なり判定にはデータ加工が必要となる。以下対応案に記載する２パターンに対し、検証を行い、重なり判定の精度を確認した。

1）対応案

任意座標との重なりを判定するため、以下２つの方法を提案する。検討内容の詳細は「別紙_8-2_登記所備付地図の任意座標系」を参照するものとする。

案①

任意座標系の住所をジオコーダーで座標に変換し、検索座標との距離で判定

案②

検索座標を逆ジオコーダーで住所に変換し、任意座標系の住所と一致判定

2）検証方法

ランダムに取得した 10 件の検索地点で対応案①②を比較する。

案①

・国土地理院のジオコーダーを使用

・検索座標との距離を計算し、50m 以内を「重なりあり」と判定

案②

・東京大学提供の逆ジオコーダー（jageocoder)を使用

・任意座標系の住所と完全一致した場合「重なりあり」と判定

3）**検証結果**

	案①（距離比較） 10 件中 7 件一致	**案②**（住所比較） 10 件中 2 件一致
メリット	・一致率が高い ・距離閾値を調整することで柔軟な判定が可能	・厳密な住所一致判定が可能
デメリット	・ジオコーダー精度に依存 ・誤判定（近隣建物が近い場合、重なり判定される） ・地番レベルの精度が保証されない	・住所表記揺れや、逆ジオコーダー精度に依存 ・誤判定（別地番になる） ・地番レベルで取得できないケースあり

表 8.2-1 検証結果

案①の距離比較については、10 件中 7 件一致し、案②の住所比較については、10 件中 2 件一致する結果となった。検証結果の詳細は「別紙_8-2_登記所備付地図の任意座標系の重なり判定方法検証」を参照するものとする。

4）**考察**

検証結果から、案①は一致率が高く、距離閾値を調整することで柔軟な判定が可能である一方、ジオコーダー精度に依存し、近隣建物を誤判定するリスクがある。案②は厳密な住所一致が可能だが、一致率が低く、住所表記揺れや逆ジオコーダー精度に影響されやすい。

今後の運用に向けた検討ポイントとしては以下が挙げられる。

・大量データ処理や近接性を重視する場合 ⇒ 案①が適切
　座標同士の距離計算のみで、住所の文字比較や表記揺れ補正が不要。
・精度を重視する場合 ⇒案②を補完的に利用
　住所の完全一致により誤判定を減らすことができる
・両手法を組み合わせるハイブリッド判定
　距離判定で候補を絞り、住所一致で精度を担保する組み合わせが有効

8.2.2 検索時の住所の表記揺れに関する課題

現行ロジックでは、登記所備付地図データを DB 検索する際に検索座標の住所を検索キーとし、前方一致で絞り込みを行っている。検索座標から住所を生成する際は、逆ジオコーダーで市区町村コードを取得し、不動産ライブラリの「都道府県内市区町村一覧取得 API」を利用して市区町村名を取得している。しかし以下 2 点の課題が確認されたため、対応策を検討する。

1）課題

A） 住所の表記揺れ

登記所備付地図データと逆ジオコーディング結果で町名に表記揺れがあり、検索時にヒットしない事象が確認された。

例：「麹町」（登記所備付地図のデータ）と、「麴町」（逆ジオコーディング）。

B） 検索キーに市名が抜けている事象

都道府県内市区町村一覧取得 API が区名のみを返却することが判明した。これにより例として「横浜市鶴見区」ではなく「鶴見区」のみの取得となる。結果として、生成された住所は「神奈川県鶴見区豊岡町」となり、前方一致検索でヒットしない状態となる。ただし、東京都 23 区など市名が存在しない場合はこの事象は発生しない。

また、不動産ライブラリの不動産価格（取引価格・成約価格）情報取得 API の大字レベルでの住所の絞り込みでも同様の住所生成ロジックを使用しているため該当データが取得できていない。

2）対応策

現行の住所文字列の一致による絞り込みは、外字や表記揺れにより精度が低下する課題がある。これを解決するため、住所文字列の一致ではなく、市区町村コードのみでの絞り込みが有効だと考える。この方法を導入することで、表記揺れの問題に加え、市名抜けの問題も解消できる。市区町村コードで絞り込みをすることで API を介した市区町村名取得が不要となり、より精度の高い検索が可能になると考える。なお、市区町村コードでの絞り込みに関する詳細なロジックや精度検証は次年度の対応とする。

8.3 PLATEAU API への要求仕様

8.3.1 検索対象用のポリゴン情報の追加

要望として、CityGML データに検索対象として利用できるポリゴン情報（例：平面ポリゴン）を追加していただきたい。これにより、現行の底面ポリゴン抽出処理を省略または簡略化し、システムの負荷と処理時間を大幅に軽減できる。

1）現状と課題の整理

- 現状

  PLATEAU の CityGML データを 1 地物単位に分割し、LOD1 情報から最小 Z 値を抽出して底面ポリゴンを生成している。全ファイルに対して抽出処理を実行しているため、処理時間が長くなっている。

- 課題

  ・全ファイルに対して底面ポリゴン抽出処理を実行しているため、処理に時間を要している。

  ・1 つの地物（建物、道路、洪水推定区域など）に対して、複数のポリゴンから構成されている場合、どの情報を基準とするか検討が必要。建物の場合、面（壁・屋根・床など）ごとに複数ポリゴンが存在するため、現状は複数ポリゴンから底面になるポリゴン情報を抽出して、基準としている。

  ・接頭辞によっては抽出した底面ポリゴンが LOD1 に対して極端に小さい範囲が抽出されるケースがある。

  ・地物やエリアごとに LOD レベルや保持する属性情報が異なるため、データ取得にあたって、LOD レベルの優先順位の決定、取得対象となる属性情報の整理、属性や LOD に応じた抽出方法の検討が必要となる。

  これらの課題に関連し、現行の底面ポリゴン抽出処理の適用状況と問題点を整理するための調査を実施した。調査結果と次年度の改善方針については、8.3.1 の 2)PLATEAU の底面ポリゴンの見直しで示す。

- 期待結果

  検索対象用のポリゴン情報が追加されることで、複数ポリゴンから基準ポリゴンを選定する必要がなくなり、さらに底面ポリゴン抽出処理を省略または簡略化が可能となるため、全体の処理時間短縮やシステムの負荷の軽減が期待できる。また、ポリゴン情報を提供していただく際には、属性情報の統一も併せて検討していただきたい。属性が統一されていれば、LOD1 以外（LOD2,LOD3 など）からでもポリゴン情報を容易に取得可能となる。

2）PLATEAU の底面ポリゴンの見直し

8.3.1 で示した課題に関連し、洪水推定区域（fld）や津波浸水想定（tnm）などの区域データで底面ポリゴン抽出結果が極端に小さくなる事象が確認されたほか、区域系のデータを 1 地物単位に分割すると、穴あきのような形状となっており、正しい区域データを保持できていないことが判明した。そのため、検索座標と PLATEAU から取得したデータの重なり判定において、期待するデータの取得ができていない状態となっている。これらの事象を踏まえ、抽出処理の精度や方法を見直す必要があると判断し、現状仕様の問題を整理し、次年度に向けた改善方針を検討することを目的として調査を実施した。課題の詳細は別添の「別紙_ 8-3-1_PLATEAU の底面ポリゴン抽出の課題」を参照するものとする。

- 調査内容

  まず、各接頭辞とそれぞれの LOD 構造の理解を深めるため、接頭辞ごとの説明と LOD ごとの形状を表にまとめた。詳細は別添の「別紙_ 8-3-1_PLATEAU の接頭辞ごとの形状」を参照するものとする。この整理により、接頭辞ごとに形状が異なることが判明したため、さらに LOD1 に着目し、現状の仕様でどのように底面ポリゴンが抽出されているか、形状による特性を分析した。詳細は別添の「別紙_ 8-3-1_各接頭辞の底面ポリゴンの抽出調査」を参照するものとする。

- 調査結果

  形状タイプごとの底面ポリゴン抽出の現状と課題は以下の通りである。詳細は別添資料の「別紙_8-3-1_接頭辞ごとの底面ポリゴン調査結果一覧表」を参照するものとする。

  A） 立体 3D（bldg,veg）

  建築物モデル（bldg）や植生モデル（veg）は閉じた立体構造を持っており、1 地物単位で完結しているため、現状の抽出ロジック（最小 Z 面抽出）

で底面ポリゴンを正しく取得できている。ただし、建築物モデル（bldg）ではユースケースにより周辺情報は不要だが、植生モデル（veg）は一帯に植物が存在しているかを示す情報であり、地物が点在しているため、検索座標との単純な重なり判定だけでは不十分である。今後は、検索座標の周辺に地物が存在するかを確認し、半径〇m 以内に該当する地物情報を取得する方針が望ましい。

B) 面 2D（tran, luse, lsld, urf）

現在の仕様で底面ポリゴンは抽出できている。しかし、LOD1 情報は基本的に Z 値が 0 の XY 平面ポリゴンで構成されるため、底面抽出は不要となり、検索座標との重なり判定は Z 値を無視した XY 平面への投影で行う方針が適切と考える。また、交通モデル(tran)や土地利用モデル(luse)は 1 地物単位では範囲が小さく、検索範囲全体をカバーするには不十分である。区域系データ（lsld, urf）は 1 地物のみだと区域情報が欠落することが多く、正しい区域情報を保持できないため 1 地物のみでなく周辺情報も必要となる。

C) 面 3D（fld,htd,frn）

洪水推定区域（fld）や高潮浸水区域（htd）は高さを持つ三角形群で構成されており、現状の最小 Z 抽出では三角形 1 枚のみが選択され、範囲が極端に小さくなっていることが判明した。また、1 地物のみでは区域情報が欠落するため、周辺情報も必要となる。都市設備モデル（frn）も同様に、面 3D の場合は現状のロジックでは底面抽出が困難である。さらに、都市設備は街路灯やベンチ、花壇などの設備で構成され、1 地物が点在しているため、検索座標との単純な重なり判定だけでは不十分である。今後は、検索座標の周辺に地物が存在するかを確認し、半径〇m 以内に該当する地物情報を取得する方針が望ましい。

D) TIN 型（dem）

地形モデルは三角形群で構成されており、面 3D 同様に現状の最小 Z 抽出では三角形 1 枚のみが選択され、底面ポリゴンを抽出できない。ただし、区域系データのように 1 地物が部分的な範囲の構造ではないため周辺情報は不要だと考えられる。

- 次年度対応方針

調査結果から、現状の底面ポリゴン抽出ロジックは形状タイプによって適用方法や精度が異なることが判明した。また、検索座標に対して周辺情報を考慮する必要性は接頭辞が示す情報の性質によって左右される。次年度は、以下の課題に対して改善方針を検討する。

A) 立体 3D（bldg,veg）

【課題】

・植生モデル（veg）は地物が点在しており、重なり判定だけでは情報を見逃す可能性がある。

【対応方針】

・検索座標に対して周辺情報を考慮する仕組みの導入を検討する。例えば、底面ポリゴンを抽出して半径〇m 以内の情報を抽出する。

B) 面 2D（tran,luse,lsld,urf）

【課題】

・LOD1 情報は Z 値が 0 の XY 平面で構成されるため、底面抽出処理は不要。

・区域系データは 1 地物単位では範囲が不十分で、情報欠落の可能性がある。

【対応方針】

・底面ポリゴンではなく LOD1 情報をそのまま Z 値を無視した XY 平面への投影として抽出する。

・検索範囲に応じた複数地物の統合処理を検討する。

→CityGML ファイル内の地物を統合する（ファイル単位）

C) 面 3D（fld,htd,frn）

【課題】

・現行ロジックでは三角形 1 枚のみ抽出され、範囲が極端に小さい。

・都市設備モデル（frn）は地物が点在しており、重なり判定だけでは情報を見逃す可能性がある

【対応方針】

・底面ポリゴンではなく LOD1 情報をそのまま Z 値を無視した XY 平面への投影として抽出する。

・検索範囲に応じた複数地物の統合処理を検討する。

・1 地物が点在している接頭辞に関しては、検索座標に対して周辺情報を考慮する仕組みの導入を検討する。例えば、底面ポリゴンを抽出して半径○m 以内の情報を抽出する。

D) TIN 型（dem）

【課題】

・現行ロジックでは三角形 1 枚のみ抽出され、範囲が極端に小さい。

【検討方針】

・底面ポリゴンではなく LOD1 情報をそのまま Z 値を無視した XY 平面への投影として抽出する。

E) 共通検討事項

・不動産ライブラリで取得可能な情報は PLATEAU 側では除外する。来年度は不動産ライブラリとの重複領域を精査。

・本調査は各形状でそれぞれ 2,3 個の接頭辞を選別し調査をして周辺情報の必要性を確認したが、全接頭辞の情報特性を整理し、周辺情報の必要性を判別。

・地物を分割せずにファイル単位で保持する方式、または分割後に統合する方式の検討。

・本調査は LOD1 のみに着目しているが、LOD1 以外（LOD2、LOD3 など）の特性や取得方法の精査。

8.3.2 地番検索の追加

要望として、API で地番検索機能の追加を検討いただきたい。これにより、検索対象をより詳細に絞りこみ、不要なデータ取得を回避できるようにしたい。

1）**現状**

検索座標の緯度経度をズームレベル 18 の XYZ タイル座標に変換し、そのタイル情報をパラメータとしてダウンロードファイル URL を取得している。

2）**課題**

現行では、タイル範囲に含まれる不要なデータまで取得してしまうため、処理に時間がかかっている。検索座標と、各データが保持するポリゴン情報との重なりを判定する際のパフォーマンスを向上させるため、対象データを事前に絞り込む必要がある。

3）**期待効果**

地番検索を可能とすることで以下の改善が期待できる。

・必要なデータのみ取得可能となり、タイル検索に比べて取得データを最小限に抑えられる。
・対象データを事前に絞り込めるため、負荷や処理時間が軽減する。

8.3.3 ダウンロードファイル取得 API の整備

要望として、ファイル URL リスト取得 API（/datacatalog/citygml/{conditions}）において、空間 ID（z/x/y）でリクエストする際、すべての地域で Z=18 の XYZ タイル座標に対応した情報を取得できるように整備していただきたい。

1）**現状**

現行は PLATEAU の API にてファイル URL リストは空間 ID（z/x/y）を指定して取得している。検索座標を Z=18 固定で XYZ タイル座標に変換し、その値を空間 ID として使用している。

2）**課題**

① Z=18 の XYZ タイルで API から情報を取得できない事象が確認された。現在確認されているのは「岡山県岡山市北区周辺」と「福島県相馬市周辺」であり、その他にも Z＝18 で取得できない地域があると考える。これらの地域ではデータ自体は存在しており、PLATEAU 公式画面は取得が可能であることを確認している。

② 「岡山県岡山市北区」周辺では Z＝18 では取得できなかったが、Z=13 以下では結果が返却された。しかし、こちらの返却結果は「倉敷市」の情報であり「岡山市」の情報は含まれていなかった。また、z=13 の結果は htd や dem などの広範囲データ（2 次メッシュ）のファイルのみの情報しかなく、bldg などの 3 次メッシュで整備されている情報は取得できなかった。

これらの課題から API 側で Z=18 の空間 ID に対応する情報の整備をして、正確なデータ返却を可能にしていただきたい。

8.4 バッチ完了通知パターン案

不動産ライブラリは、リクエストを受信時に取得を行い、PLATEAU、登記所備付地図については、非同期のバッチ処理にて取得を行う。取得・変換の完了タイミングはそれぞれ異なる。

また、PLATEAU、登記所備付地図については、既に取得済みで短期インデックスにデータが存在した場合、バッチ処理を実行せず、短期インデックスから取得が可能である。そこで、通知を行うタイミングとしては、以下のタイミングが想定される。

パターン	タイミング	バッチ実行	返却可能データ			備考
			不動産	PLATEAU	登記所	
1-1	不動産ライブラリ取得完了時	必要	○	×	×	
1-2		不要	○	○	○	
2	PLATEAU 取得完了時	-	○	○	△	
3	登記所備付地図 取得完了時	-	○	△	○	

表 8.4-1 通知タイミングパターン

パターン 1-1、1-2 は同じタイミングではあるが、バッチ実行の要否によって、返却できるデータが異なる。パターン 2 のタイミングで、登記所備付地図の取得バッチが完了していない場合、登記所備付地図のデータは返却できない。パターン 3 はその逆で、PLATEAU の取得バッチが完了していない場合、PLATEAU のデータは返却できない。バッチ処理は時間が掛かることから、全てのデータが取得済みとなったタイミングで通知を行うと、ユーザの待ち時間がそれなりに掛かる。各パターンで取得済みのデータを返却する形にすれば、通知が複数回にはなるが、ユーザは早いタイミングから内容を確認することができる。

プロトタイプツールでは、上記の 4 パターン、回数としては最大 3 回の通知を行うようにしている。

8.5 データ返却パターン案

プロトタイプツールでは 3 パターンの方式を検討した結果、抽出された各種ファイルを ZIP 形式にまとめて返却する方式を採用している。これは、可視化ツールでの利用を前提としたものである。しかし、今後の展開に向けて以下のような出力形式および返却方法について検討する必要がある。3 パターンの方式検討の詳細の内容については、第 4 章の 4.4 データ出力方式の検討を参照するものとする。

1）GeoJSON 形式への統一

PLATEAU の CityGML データを GeoJSON 形式に変換し、不動産ライブラリ API の GeoJSON と形式を揃える。複数の GeoJSON ファイルを統合し、1 つの GeoJSON ファイルにする。

【メリット】

- 可視化ツールとの親和性が高い。
- 同じ形式にそろえることで、フロントエンドや可視化ツール側の実装が容易となる。
- 1 ファイルにすることで管理が簡素化される。

【デメリット】

- ポリゴンの視認性が低い。
- CityGML の詳細な属性情報が GeoJSON 形式に変換することによって失われる可能性がある。
- 結合後のファイルサイズが大きくなる場合がある

【ユースケース】

- 地図上に建物や土地情報を 2D で表示する用途。ユーザがブラウザ上で直感的に情報を確認できる。

2）GeoPackage 形式による統合返却

複数の空間レイヤや属性情報を 1 つの GeoPackage ファイルにまとめて返却する方式。

【メリット】

- 複数のレイヤや属性情報を 1 ファイルに格納できるため、ファイル管理や配布がシンプルになる。
- QGIS などの主要な GIS ツールで直接読み込み可能で、空間解析や編集が容易。
- 属性情報の構造を比較的保ちやすい。

【デメリット】
- PLATEAU の CityGML が持つ 3D 構造は GeoPackage では正確に再現できない可能性がある。
- Web 地図ライブラリでは直接扱えないため、別途変換が必要。
- 専用ソフトが必要となる（QGIS など）。

【ユースケース例】
- GIS ソフトを用いた空間分析。
- インターネット接続がない環境でも、ローカルで完結した作業が可能。
- 複数レイヤを 1 ファイルで管理したいケース

3）CSV 形式による返却

緯度・経度やポリゴン情報を含めた CSV 形式で出力。

【メリット】
- CSV はシンプルで、Excel や統計ツールで容易に利用できる。
- Excel などで直接開いて編集できるため、非技術者にも扱いやすい。
- 座標や属性を選択的に保持することで、データサイズを削減できる。

【デメリット】
- 緯度経度やポリゴン情報を持たせることは可能だが､複雑なジオメトリや LOD 情報は保持できない。
- GIS や 3D 解析ツールでの互換性が低く、再利用性が下がる。
- 可視化ツールで表示するには追加処理や操作が必要。

【ユースケース例】
- 地理的な構造よりも、統計分析や属性抽出が中心となる場面。
- 形式でのデータ提出や、Excel ベースの業務処理。

4）ZIP 形式によるファイル単位返却（現行方式）

出力された各ファイル（GeoJSON, CityGML 等）をそのまま ZIP 形式にまとめて返却する方式。

【メリット】
- 元データの構造を保持できる。
- ユーザ側で柔軟に加工や選別が可能。
- GeoJSON は可視化、CityGML は 3D 解析など、目的に応じた使い分けが可能。

【デメリット】
- 可視化ツールや AI 処理に使うには、前処理や操作が必要になる場合がある。
- ファイル数が多いと管理が煩雑になる

【ユースケース例】

・　利用目的が多岐に渡る可能性がある場合。
・　形式ごとの特性を活かした処理を行いたいケース。

返却方法については、ユースケースに応じた複数の選択肢が存在する。ニーズとデータの性質を踏まえた最適な返却方法の検討が必要である。

8.6　不動産 ID の導入検討

8.6.1　各業界の不動産 ID に対するニーズ

第 7 章で実施した有識者ヒアリングにおいて、連携環境へのニーズだけでなく、不動産 ID の導入に対する期待や具体的なニーズが複数確認された。本節では、各業種から寄せられた不動産 ID へのニーズを整理する。

1）宅地系不動産業・デベロッパー業

不動産やデベロッパーの価格査定業務において、建物の建替え履歴や改修履歴を追跡できることへのニーズが確認された。物件の価格査定を行う際、建物の過去の情報が重要な判断材料となる。しかし、現状では以下のような課題がある。

建築年の正確な把握が困難	登記情報には新築時の建築年は記載されているが、その後の大規模改修や増築の情報は必ずしも記載されていない
改修履歴の欠如	耐震補強、外壁改修、設備更新等の改修履歴が体系的に記録されていない
建替え前後の関係性が不明	同一敷地で建物が建て替えられた場合、建替え前の建物との関係性が明確でない
過去の取引事例との紐付けが困難	住所ベースでの検索では、建替え前の取引事例と建替え後の建物を関連付けることが難しい

表 8.6-1 宅地系不動産およびデベロッパー業の業務課題

これらの課題により、査定の精度が低下し、適正な価格評価が困難となっている。

【不動産 ID 導入による期待効果】

不動産 ID が導入され、建物の履歴情報が体系的に管理されることで、以下の効果が期待される。

1	建物履歴の時系列追跡	不動産 ID をキーとして、建物の新築から現在までの全ての履歴を追跡可能に
		改修履歴（耐震補強、外壁改修、設備更新等）を時系列で把握
		用途変更の履歴（住宅→店舗、事務所→住宅等）も記録
2	建替え前後の関連付け	建替え時に、新しい不動産 ID が付与されると同時に、旧不動産 ID との関連情報も記録
		同一敷地における建替え前後の建物を容易に特

		定可能
		敷地の分割・合筆の履歴も追跡可能
3	過去の取引事例との紐付け	不動産 ID をキーとして、過去の取引事例を検索可能
		建替え前の建物の取引価格も参照し、土地の価値を評価
		同様の改修を行った建物の取引事例を参考にした査定が可能
4	査定精度の向上	建物の状態（築年数、改修状況）を正確に把握した上での査定
		類似条件の建物の取引事例を効率的に検索
		より根拠のある、説得力のある査定価格の提示
5	リスク管理の強化	過去の災害被害履歴（浸水、地震被害等）を把握
		建物の構造や耐震性能の変遷を追跡
		適切なリスク評価と保険料算定

表 8.6-2 不動産 ID 導入による期待効果

2）保険業

某損害保険会社へのヒアリングにより、法人向けの火災保険業務において、同一敷地内に複数建物が存在する工場や商業施設に対して、建物ごとに ID を付番できることへのニーズが確認された。現状は以下のような課題がある。

同一敷地内の複数建物の識別困難	工場の敷地内に、本館、別館、倉庫、ボイラー室等、複数の建物が存在する場合、これらを一意に識別することが困難
個別のリスク評価ができない	建物ごとに構造、用途、設備が異なるため、本来は個別にリスク評価すべきだが、現状では敷地全体としての評価になりがち
保険契約管理の煩雑さ	建物の増築、取り壊し、用途変更等があった場合、どの建物に対する変更かを特定することが困難
座標だけでは不十分	近接する複数の建物を座標だけで区別することは難しく、住所や建物名称に頼らざるを得ない
建替え・改修履歴の欠如	固定資産台帳だけでは、建物の建替え履歴や改修履歴が分からない

表 8.6-3 保険業の業務課題

これらの課題により、適切なリスク評価ができず、保険料の算定精度が低下している。また、事故発生時に、どの建物が対象かを特定するのに時間を要することもある。

【不動産 ID 導入による期待効果】

不動産 ID が導入され、同一敷地内の各建物に個別の ID が付与されることで、以下の効果が期待される。

1	建物の一意識別	同一敷地内の各建物（本館、別館、倉庫等）に個別の不動産 ID を付与
		住所や建物名称によらず、ID で一意に識別可能
		建物の増築、取り壊し等の変更管理が容易に
2	建物ごとの個別リスク評価	各建物の構造（耐火、準耐火、木造等）を正確に把握
		各建物の用途（事務所、工場、倉庫等）に応じたリスク評価
		各建物に設置されている設備（消火設備、防火設備等）の情報を個別に管理
		建物一つ一つのリスクを算出し、適切な保険料を算定
3	保険契約管理の効率化	不動産 ID をキーとして、各建物の保険契約を管理
		建物の増築、取り壊し、用途変更等があった場合、該当する不動産 ID の情報を更新
		契約内容の変更、更新手続きが効率化
4	事故対応の迅速化	事故発生時に、不動産 ID により対象建物を即座に特定
		該当建物のリスク情報、保険契約内容を迅速に参照
		損害査定、保険金支払いの迅速化
5	建替え・改修履歴の活用	不動産 ID に紐づく建替え履歴、改修履歴を参照
		耐震補強、防火設備の更新等の改修により、リスクが低減している場合は保険料を調整
		より公平で精緻な保険料算定が可能
6	連携環境との統合	不動産 ID をキーとして、連携環境から各建物周辺のハザード情報を取得
		全国に点在する施設・工場の災害リスクを一括評価
		リスクベースの保険料算定の高度化

表 8.6-4 不動産 ID 導入による期待効果

【ヒアリングで得られた具体的なニーズ】

・ 「工場のように、敷地内に複数の建物がある場合、建物ごとの情報（建替え履

歴/改修履歴）を取得したい」
- 「同敷地内の建物ごとに、個別に不動産 ID を付与してほしい」
- 「不動産 ID があれば、建物一つ一つのリスク算出ができ、より適切な保険料算定が可能になる」

8.6.2 連携環境における不動産 ID の活用イメージ

本調査で開発したプロトタイプツールは、現時点では座標を入力インターフェースとしているが、将来的に不動産 ID が整備された際には、不動産 ID を用いたデータ取得も可能とすることが望ましい。

本節では、連携環境において不動産 ID をどのように活用するかを検討する。なお、不動産 ID の整備を行っている国土交通省不動産市場整備課と打ち合わせを行い、不動産 ID の想定仕様を確認したうえで、データ連携環境との合流方式を検討した。

1）不動産 ID によるデータ取得の基本的な流れ

不動産 ID を用いたデータ取得は、以下の流れで実現することを想定する。

① ユーザが不動産 ID を入力
② 不動産 ID から基本情報を取得（座標、住所、建物種別等）
③ 取得した座標・住所を用いて、各データソースから関連情報を取得
④ 不動産 ID に紐づく履歴情報を取得（建替え履歴、改修履歴、取引履歴等）
⑤ 統合されたデータを返却

2）座標指定と不動産 ID 指定の併用

連携環境では、座標指定と不動産 ID 指定の両方をサポートすることが望ましい。座標指定の場合、不動産 ID に登録されている座標点が指定座標点と一致しない可能性が高いため、以下の方法でマッチングを実施する。

【座標指定の場合】

① ユーザが座標を指定
② 座標から近傍の不動産 ID 候補を検出
③ 候補が複数ある場合、住所情報でマッチング
④ 特定された不動産 ID に紐づくデータを取得

【不動産 ID 指定の場合】

① ユーザが不動産 ID を指定
② 不動産 ID から座標・住所を取得
③ 取得した座標・住所を用いてデータを取得

3）座標と不動産の紐づけ方法

ユーザが指定する座標点と不動産 ID を紐付ける際、以下の課題と対応方法を検討する必要がある。

【課題】

A) ユーザが指定する座標点と、不動産 ID に登録されている座標点が完全一致しない可能性がある
・GPS の誤差
・地図上でのクリック位置のずれ
・建物の代表点の定義の違い（重心、入口、角等）

B) 同一座標に複数の不動産 ID（階層構造）が存在する可能性がある
・同一建物内の複数の部屋
・複数の建物が近接している場合

【対応方法】

1. 座標による近傍検索
ユーザが指定した座標の近傍（例：半径 50m 以内）に存在する不動産 ID を候補として抽出する。
①入力座標:（緯度，経度）
↓
②近傍検索: 半径 50m 以内の不動産 ID を抽出
↓
③候補: 不動産 ID1，不動産 ID2，・・

2. 住所情報によるマッチング
候補が複数ある場合、住所情報を用いてマッチングを行う。
①入力座標から逆ジオコーディングで住所を取得
↓
②各候補の不動産 ID から住所情報を取得
↓
③住所の一致度を計算（前方一致、部分一致等）

↓

④最も一致度の高い不動産 ID を選択

3. ユーザ確認

候補が複数あり、自動判定が困難な場合は、ユーザに選択を促す。

①複数の候補を提示（不動産 ID、住所、建物種別、階層等）

↓

②ユーザが該当する不動産 ID を選択

8.7 システム性能、アーキテクト関連の検討

8.7.1 プロトタイプのバッチ処理部の性能評価

プロトタイプツールの作成と検証にあたり、変換処理の責務を分割し、AWS の Lambda でそれらを実装した。処理結果は S3 に保存し、後続の変換処理を S3 のイベント駆動で連携する方針とした。

加えて、PLATEAU データ、登記所備付地図データを取得後、そのファイルを地物単位に分割し、別ファイルとして S3 に保存している。処理をファイル単位で並列に実行することで、処理時間の短縮を図った。AWS 側の設定で Lambda の同時実行数（デフォルトで 1000、今回 2000 まで引き上げ）を調整し、最大 2000 の同時実行で行った。

このような方針で作成したプロトタイプツールを検証したところ、以下の問題点が浮上した。

1）S3 への保存がボトルネック

PLATEAU データ、登記所備付地図データを取得後、地物単位に S3 に保存する処理はシーケンシャルに行う必要がある。S3 への保存は、AWS 内の API コールとなり、オーバーヘッドが大きい。1 ファイル内の地物が多いと S3 へ保存する処理の比率が高くなる。例えば、PLATEAU の世田谷区など都心部のデータは、1 ファイルに 5 万件ほどの地物が存在しているものがあり、ファイルサイズも 100MB を超える。

おおよそ地物が 5000 件程度で、Lambda の実行時間上限の 15 分に到達し、処理を完了することができなくなった。プロトタイプツールでは、暫定的に 5000 件程度で、一度 Lambda の処理を終了し、同じイベントを発生させ、ファイル分割を継続して行うよう対応した（本来、Lambda でこのような利用は推奨されていない）。

結果的に、全体の処理時間のうち、S3 へ保存する時間の比率が高くなった。変換処理以外にコンピュートリソースを割くことになり、過分な課金額が発生することとなった。

2）**短期インデックスへの登録がボトルネック**

短期インデックスは、変換後のデータを短期にキャッシュする。データを管理する用途ではないので、データ管理で使用されるリレーショナル DB は不向きとなる。また、リレーショナル DB は、同時コネクション数の制限が強い。変換処理の同時実行数を増やし処理時間を短縮する方針では、短期インデックスへの登録で、コネクション数が枯渇することが検討初期より予想されていた。当然、性能の高い DB を用意すれば、最大同時コネクション数を上げることはできる。一方、検索時に利用するリソースとしては過多になり、課金額の採算が取れなくなる。

上記のデメリットがあるが、プロトタイプツールの構築に、短期インデックスとしてリレーショナル DB を用いた。主な理由は、地理空間の距離で検索ができるマネージドサービス（AWS の RDS for PostgreSQL）として構築が容易であることが挙げられる。

参考値として、vCPU が 4、メモリが 16G の RDS for PostgreSQL にて、デフォルト（AWS での推奨値）の最大コネクション数は 400 となる。よってこの性能の RDS では、最大同時実行数が 2000 の Lambda が同時に短期インデックスへ登録を行おうとするとコネクションが枯渇し、登録処理が停滞することになる。

8.7.2 バッチ処理部のリアーキテクト

プロトタイプツールで採用した、各変換処理を独立させる方針は、連携のために用いる S3 への登録がオーバーヘッドになる。一連の処理を 1 つの ETL（抽出・変換・ロード）処理として構築し、途中結果は S3 へ保存しない方が好ましいと言える。

また、ETL 処理の最後に短期インデックスへ登録するが、その際のコネクションも 1 つ用いればよく、コネクションの枯渇への対策にもなる。

具体的には、PLATEAU、登記所備付地図向けに 2 つの ETL を定義し、取得のキー（PLATEAU の場合メッシュコード、登記所備付地図の場合、市区町村名）にてパイプランが実行されるようにする。

ETL のコード記述のスタイルは採用する ETL のフレームワークに依るが、データの取得、変換部分のコードは、プロトタイプツールで作成したものが流用できる。

8.7.3 プロトタイプツールで対象外とした機能

プロトタイプツールでは、PLATEAU、登記所備付地図のデータについて、短期インデックスに存在しなければバッチ処理をフックするよう簡易的な実装した。

ただ短期インデックスを検索する際、ポイントデータについては指定半径のもののみを取得する。実際には取得済みであるが、指定半径に存在しない場合、バッチ処理が無駄にフックされてしまう。バッチ処理は処理コストが高いため、厳密にハンドリングする必要がある。

PLATEAU についてはメッシュコード、登記所備付地図では市区町村名がデータ取得のキーになる。このキーで、取得中／取得済みであるかのステータス管理を行い、その管理情報を参照してバッチ処理をフックするかをハンドリングする機能が必要となる。

ステータスを管理するので、採用するデータストレージとしては、ACID 特性を備えたものが必要となる。

8.7.4 短期インデックスの要件

データ管理でなく、変換後のデータを検索することが主要な用途となるため、DB より検索エンジンに分類されるソフトウェアが適していると言える。

検索エンジンの多くは、新規登録と更新を区別せず、データを登録するシンプルなオペレーションになっているのも特徴と言える。

また、プロトタイプツールでは、不動産ライブラリ、PLATEAU、登記所備付地図について、それらを取得・変換後、短期インデックスへ登録している。それらの処理が完了したタイミングで、短期インデックスから、指定座標の半径内に収まっているポイントデータやポリゴンが重なっているデータを抽出し、返却する仕組みとなっている。よって、検索エンジンには地理空間での検索が可能なものが必要となる。

短期インデックスは、変換後の一時的なキャッシュであるため、有効期限が設定でき、期限が切れたデータは自動で削除される仕組みが備わっている方が良い。

利用する利用ユーザ数の規模を予め想定することが難しいため、規模が大きくなっても問題なく稼働するスケール耐性が備わっているものが良い。

以上から、次に挙げる要件が必要となる。

- リレーショナル DB のような ACID 特性は不要（Should）
- 新規登録と更新を区別せずデータを登録できる（Should）

- 地理空間での検索が可能（MUST）
- 有効期限があり、期限切れのデータは消去される（MUST）
- ユーザ数の増加に伴い、データが増えるのでスケール耐性がある(Should)

代表的な DB、検索エンジンについて、上記の要件を満たしているか表として纏めた。

	ACID 特性	データをセット	地理空間検索	有効期限	スケール耐性
Elasticsearch	×	○	○	○	○
MongoDB	×	○	○	○	○
PostgreSQL	○	△(※1)	○(※3)	×	△(※2)
MySQL	○	△(※1)	○	×	△(※2)

表 8.7-1 要件の対応表

※1 SQL の記述で対応できるが、内部的には一度 insert を試み、できなかったら update に切り替える方式
※2 パーティション分割（PostgreSQL）やリードレプリカ（PostgreSQL、MySQL）を用意して負荷を分散する方式
※3 追加の機能拡張を導入する必要がある

スケール耐性については、NoSQL に分類されるクラウドのフルマネージドサービスが有益な場合がある。これらのサービスでは地理空間による検索はできない事が多い。しかし、総務省の地域メッシュ（PLATEAU のメッシュコードがこれに該当）や geohash, quadkey といったハッシュ値を元にデータを短期インデックスへ登録し、検索時にはハッシュ値を使用して取得する。取得データがポイントデータの場合、指定緯度経度から半径内に存在しているか、ポリゴンデータの場合、重なりがあるかをプログラム上のロジックでフィルタすることで実現可能である。

クラウドのこれらのサービスは、データのオペレーションに対しての従量課金であることが多い。利用頻度が低い場合には課金のメリットがあるが、利用頻度が高くなるにつれ、コンピュートノード単位での課金の方がメリットとなる分岐点が存在する。

以下、各クラウドの NoSQL サービスにおける地理空間による検索が可能であるか、課金形態について纏めた。どのサービスでも内部的には PK によりシャーディングが行われる。それを意識した PK の設計が必要であるが、リレーショナル DB に比べ、容

易にスケール耐性を得られる。

	課金形態	地理空間検索	備考
Amazon DynamoDB	従量	×	
Azure Cosmos DB	従量	○	
Google Spanner	ノード単位	×	● SQL レベルの PostgreSQL 互換モードがあるが、地理空間検索が可能であるか調査必要
Google Firestore	従量	×	● MongoDB 互換モードがあるが、地理空間検索が可能であるか調査必要

表 8.7-2 クラウドデータベースサービスの比較

8.7.5 利用ユーザの認証の仕組み検討

プロトタイプツールでは、不動産ライブラリ、PLATEAU、登記所備付地図のデータを取得する。各サービスでは利用のためにユーザ登録が必要で、登録後、API キーを取得する。リクエストにレート制限のようなものは明示されていないが、利用規約に反する使用には、利用ユーザへの通知やユーザ登録の削除なのが想定される。

地理空間情報の連携環境として、クラウド上に処理基盤が存在するので、その基盤を利用するためのユーザ登録、利用時の認証は必要と言える。

Web 上でのユーザ登録画面、ログイン画面、ログイン後のダッシュボード画面、実際に検索リクエスト入力画面といった画面が必要となる。

1）認証の方式

各クラウドで Id Provider のサービスが存在するので、そのサービスを利用し、一般的な Web 画面の認証を構築する。もし、他の認証基盤と連携したい場合でも、それらのサービスで外部の Id Provider と連携ができるようになっている。標準化されている OpenID Connect や SAML であれば連携は容易となっている。

もし、Web 画面以外から検索リクエストを送信したい場合、Web での認証は難しい。この場合、API キーを払い出す画面を Web 上に用意し、Web 画面以外からの検索リクエストを API キーで認証する必要がある。

【ユーザの登録】

メールアドレスをユーザの識別子として、ユーザ登録を行う。メールアドレスが有

効なものであるかチェックするため、そのメールアドレスへメールを送信し、メール内のリンクから、実際の登録処理を実施する形が良い。

同様に、メールアドレスを変更する場合、ログイン済みの状態で、新しいメールアドレスを入力してもらい、そのアドレスにメール送信後、そのメール内のリンクをクリックすることで変更が完了する仕組みが良い。

2）各サービスの API キーの扱い方

不動産ライブラリ、PLATEAU、登記所備付地図のデータを取得するには、各サービスの API キーが必要となる。API キーの扱い方に、2 通りが考えられる。

① 連携環境が各サービスの API キーを保持

利用ユーザが、各サービスのユーザ登録、API キーの取得をする必要がなく、連携環境のユーザ登録だけで利用できるようになる。ただ、連携環境が内部的に各サービスを利用するので、連携環境の利用規約は、各サービスの利用規約から逸脱しないものを定義する必要がある。

各サービスで API キーによるレート制限などの制限が存在する場合、その制限に達してしまう場合が想定される。各サービスとの調整が必要になる可能性がある。連携環境で保持する API キーを複数保有し利用することで、制限に達しない仕組みが必要となる可能性もある。

② 利用ユーザが各サービスの API キーを取得し連携環境上で設定する

利用ユーザに、各サービスのユーザ登録、API キーの取得、加えて連携環境のユーザ登録を強いる形となり、煩雑ではある。ただ、各サービスの利用は利用ユーザが取得した API キーにより行われるため、連携環境は利用ユーザに代わって取得作業を行う形になる。連携環境の利用規約としては、連携環境を利用する上での規約を定義すればよい。

8.7.6 セキュリティ要件

1）クラウドのインフラレイヤーのセキュリティ

① パブリックなネットワークとプライベートなネットワークに分ける

各クラウドによってセキュリティのベストプラクティスが多少異なるが、原則、外部と通信できないクローズドなネットワークを用意し、そこにコンピュ

ートリソースを配置するべきである。外部との通信は、別のネットワークで受信し、受信するノードだけがクローズドなネットワーク内のリソースにアクセスできるようにする。
外部から直接の通信を許可しないことで、不正なアクセスをブロックする。

② クラウドのユーザアカウントとセキュリティ
運用者、開発者はクラウドを利用するためのユーザアカウントが用意される。ユーザアカウントの認証については、原則多要素認証にするべきである。パスワードが流出した場合でも、他要素の認証によってブロックすることができる。また、運用者、開発者の不用意による誤操作を防止するために、更新系のロールと参照系のロールを分け、通常は参照系のロールを使うようにすべきである。また、更新系のロールは、利用できる人間を限定するか、後述する IaC で変更の適用を管理した方が良い。人間が直接更新系の操作を行う頻度を減らすと、リスクが減る。

③ クラウドの権限について
権限は、原則ミニマムを設定するのがベストプラクティスとなる。クラウドがデフォルトで用意しているロールは使用せず、必要最低限の権限を持つロールを用意し、それをユーザアカウントや、コンピュートリソースへ付与する。

④ IaC の利用
クラウドのインフラ構築や、運用者、開発者のユーザアカウントを IaC で管理することを推奨する。2022 年の ISO 27001（ISMS）改訂で、IaC による構成管理が追加され、推奨がされている。
特にユーザアカウントは管理が煩雑になるため、IaC として管理するべきである。不要なユーザアカウントを残しておくと、セキュリティのリスクに繋がる。

2）アプリケーションレイヤーのセキュリティ

① 悪意のあるユーザによる利用の制限
登録したユーザが悪意を持っており、過分な負荷をかけ、システムをダウンさせようと試みるケースがある。通常の利用範囲を超えるケースについては、リクエストをブロックし、システムダウンの抑止、想定外のコンピュートリソースの消費を抑える必要がある。
連携環境としては、PLATEAU、登記所備付地図の取得・変換のバッチ処理に多

くのリソースが割かれる。一般的には、HTTP 通信のレート制限（同時または規定時間単位でのリクエスト数の上限）を設けるが、バッチ処理に多くのリソースを割く都合上、スロット制限（一人のユーザが、同時にバッチを実行できる上限）を設けるのが適している。

スロット制限は、アプリケーション上で仕組みを設ける必要がある。一方、HTTP 通信のレート制限は、フロントに WAF（Web Application Firewall）を構えることで実現できる。WAF は、IP アドレスによるブロックなど、運用上で必要となるブロック条件を指定できるので、インターネットにサービスを公開する場合は、必須となる。スロット制限のみを設ける場合でも、運用過程でブロックしたいIPアドレスなど発生し得るので、WAFの導入は必要と言える。