卒業論文

契約ネットプロトコルを用いた
マルチエージェントシステムによる
複合サービス推薦

指導教官　　村上 陽平 教授

立命館大学情報理工学部
先端社会デザインコース

飛澤 佑季

2025年1月9日

契約ネットプロトコルを用いた
マルチエージェントシステムによる
複合サービス推薦

飛澤 佑季

内容梗概

近年，Web サービスは生活や産業のあらゆる場面で不可欠な存在となり，複数の Web サービスを組み合わせて利用するサービス合成の重要性が高まっている．しかし従来手法では，ユーザーの目的や要求を時相論理・述語論理といった形式的表現で記述する必要があり，専門知識を持たない利用者にとって大きな障壁となっていた．

このような問題を解決するために，大規模言語モデル（LLM）を用いて，自然言語で記述された要求からそのゴールを満たす Web サービスを推論するエージェントが提案されてい．しかしながら，単一エージェント方式では，多数の API 仕様や専門知識を 1 つのプロンプトに集約する必要があり，プロンプト肥大化や入力制限により推論精度が低下しやすい．

そこで，複数の LLM エージェントを協調させるマルチエージェントシステムを導入し，エージェントに API 仕様を分散保持させることでプロンプト制約を緩和する．具体的には，契約ネットプロトコル（Contract Net Protocol: CNP）に基づき，マネージャが要求を公告し，コントラクタエージェントが担当可否と API 案を入札，マネージャが統合して合成計画を確定する．

本手法の実現にあたり，取り組むべき課題は以下の 2 点である．

推論タスク割り当て　自然言語で記述されたユーザ要求に対し，CNP に基づいてタスク分解を行う際に，要求の粒度や曖昧さに応じて分解方式をどのように選択すべきか，またその分解をマネージャ等のどの役割のエージェントが担うことが適切かを検証する必要がある．

コントラクタエージェントの組織化　CNP では，コントラクタエージェント数の増加に伴い公告・入札の通信量が増大し，スケーラビリティが低下しやすいという課題がある．このデメリットを緩和するため，コントラクタエージェントに対してどの API を割り当てるべきか，またコントラクタエージェント群のカテゴリ別階層化を検討し，組織構造の違いが入札の精度や合成結果に与える影響を検証する必要がある．

1つ目の課題に対しては，マネージャ主導・コントラクタ主導・協調型の3つのタスク分解プロトコルを比較する．マネージャ主導では，マネージャがタスク分解とカテゴリ分解を行ってタスク仕様を公告し，コントラクタエージェントは担当可否とAPI案を入札する．コントラクタ主導では，コントラクタエージェントが自身の担当API仕様に基づいてタスク分解とカテゴリマッピングを行い入札する．協調型では，マネージャがタスク分解を提示したうえで，コントラクタエージェントが分解されたタスクに対してカテゴリマッピングを行って入札する．本研究では，これら3つのプロトコルの分担の違いが合成計画の整合性および計画生成精度に与える影響を分析する．

2つ目の課題に対しては，コントラクタエージェントの組織化により解決を図る．本研究では，単一APIを担当させる設計を基本とし，APIカテゴリ単位で複数APIをまとめて担当させる設計については今後の検討課題として位置付ける．各エージェントは，割り当てられたAPIの仕様のみを参照して推論・入札を行うようにし，推薦精度に与える影響を検証する．

本研究では，提案手法の有効性を検証するため，ProgrammableWeb の複合サービスデータを用いた評価を行った．具体的には，複合サービスデータ 400 件をテストデータとし，対応する 909 件の Web API を対象として，マネージャ主導・コントラクタ主導・協調型という3種類のタスク分解方式を適用してサービス合成を実行し，各方式の計画生成精度を比較した．生成されたサービス計画と正解計画を照合し，適合率，再現率，F1 スコアを算出することでタスク分解方式の性能を評価した．また，コントラクタエージェントの組織化については単一API単位の割当を採用し，複数APIを担当させる設計の影響は今後の課題として整理した．また本研究の貢献は以下の通りである．

推論タスク割り当て　マネージャ主導・コントラクタ主導・協調型の3つのタスク分解プロトコルを比較し，ユーザ要求の粒度に応じた適切な方式を分析した。適合率・再現率・F1 値で評価した結果，本手法は単一方式を上回る性能を示した。

コントラクタエージェントの組織化　単一API単位で専門化したコントラクタ構成を用い，分散保持の有効性と推薦精度への影響を評価した．複数APIを担当させる設計については今後の課題として位置付けた．

Composite Service Recommendation Using Contract-Net-Based LLM Multi-Agent Systems

Yuki Tobisawa

Abstract

In recent years, Web services have become indispensable in all aspects of daily life and industry, and the importance of service composition—combining multiple Web services—has been increasing. However, conventional methods require users to describe their objectives and requirements using formal expressions such as temporal logic or predicate logic, which presents a significant barrier for users without specialized knowledge.

To address this problem, agents that use Large Language Models (LLMs) to infer Web services satisfying goals from requirements described in natural language have been proposed. However, in single-agent approaches, numerous API specifications and domain knowledge must be consolidated into a single prompt, which tends to reduce inference accuracy due to prompt bloating and input limitations.

Therefore, we introduce a multi-agent system that coordinates multiple LLM agents, allowing agents to distributedly maintain API specifications to alleviate prompt constraints. Specifically, based on the Contract Net Protocol (CNP), a manager announces requirements, contractor agents bid on their capability to handle tasks along with API proposals, and the manager integrates these to finalize the composition plan.

In realizing this approach, there are two challenges to be addressed:

Inference Task Allocation When decomposing tasks based on CNP for user requirements described in natural language, it is necessary to verify how the decomposition method should be selected according to the granularity and ambiguity of requirements, and which role of agent (such as the manager) should appropriately handle this decomposition.

Organization of Contractor Agents In CNP, as the number of contractor agents increases, the communication volume for announcements and bids grows, which tends to reduce scalability. To mitigate this disadvantage, it is necessary to examine which APIs should be assigned to contractor agents,

consider category-based hierarchization and aggregation of contractor agent groups, and verify how differences in organizational structure affect bidding accuracy, composition results, and communication load.

For the first challenge, we compare three task decomposition protocols: manager-led, contractor-led, and collaborative. In the manager-led approach, the manager performs task decomposition and category decomposition, announces task specifications, and contractor agents bid on their capability to handle tasks along with API proposals. In the contractor-led approach, contractor agents perform task decomposition and category mapping based on their assigned API specifications and submit bids. In the collaborative approach, the manager presents task decomposition, and contractor agents perform category mapping for the decomposed tasks and submit bids. This research clarifies which of these three protocols is suitable according to requirement types such as granularity and ambiguity, and analyzes how differences in task distribution affect the consistency of composition plans and plan generation accuracy.

For the second challenge, we seek solutions through organization of contractor agents. In this study, we adopt a design where each contractor agent handles a single API, and leave the evaluation of multi-API assignment by category as future work. Each agent performs inference and bidding by referencing only the specifications of their assigned API, and we verify the effects on recommendation accuracy and communication load.

To validate the effectiveness of the proposed method, we conducted evaluations using composite service data from ProgrammableWeb. Specifically, using 400 composite service data items as test data and targeting 909 corresponding Web APIs, we executed service composition applying three types of task decomposition methods—manager-led, contractor-led, and collaborative—and compared the plan generation accuracy of each method. By comparing generated service plans against ground truth plans and calculating precision, recall, and F1 scores, we evaluated the performance of the task decomposition methods. We adopt single-API assignment for contractor agents in this study and leave multi-API assignment as future work. The contributions of this research are as follows:

Inference Task Allocation We compared three task decomposition protocols —manager-led, contractor-led, and collaborative—and analyzed appropriate methods according to the granularity of user requirements. Evaluation results using precision, recall, and F1 scores demonstrated that our method outperforms single-method approaches.

Organization of Contractor Agents We evaluated a single-API contractor design and its impact on recommendation accuracy. The effects of assigning multiple APIs per contractor are left as future work.

契約ネットプロトコルを用いた
マルチエージェントシステムによる
複合サービス推薦

目次

第1章　はじめに

近年，サービスコンピューティングの発展により，多種多様な Web サービスを組み合わせて新たな付加価値を提供する複合サービスが盛んに構築されている。翻訳サービスと音声認識サービスを連携させたリアルタイム音声翻訳や，地理情報サービスと天気情報サービスを統合した地図表示サービスなど，個々の Web サービスでは実現できない機能を柔軟に実装できる。しかし，公開されている Web サービスのうち，複合サービスに実際に利用されているものは一部にとどまり，ユーザが膨大な候補の中から目的に適したサービスを選択することは依然として大きな負担となっている。

従来の複合サービス合成は，水平型サービス合成と垂直型サービス合成の二つのアプローチに大別される。前者は，所与のワークフローに対して各タスクを実行するのに適した Web サービスの組み合わせを選択する枠組みであり，ワークフロー設計のコストが高いという課題を持つ。後者は，人工知能のプランニング技術を用いて，論理式で与えられたゴール状態に到達するサービス実行系列を生成する手法であるが，ユーザが時相論理や述語論理といった形式的表現で要求を記述しなければならず，エンドユーザにとって扱いやすいとは言い難い。この問題を軽減するため，自然言語で記述された要件定義からサービスの組み合わせを推定する手法や，WSDL やサービス説明文，サービスネットワークをテキストマイニングやグラフ埋め込みによって特徴ベクトル化し，機能ごとにクラスタリングする手法が提案されてきた。

一方，近年急速に発展している大規模言語モデル（Large Language Model; LLM）は，自然言語で記述された要求から計画を生成し，外部ツールや API を呼び出す能力を備えており，ユーザに形式的な記述を要求することなくサービス合成を行う枠組みとして期待されている。しかし，単一の汎用 LLM エージェントに広範なサービス領域を一括して扱わせる場合，個々のサービス仕様や実行環境に関する専門的な知識の理解が浅くなりやすい。また，多数の API 仕様や利用例を一つのプロンプトに詰め込む必要があるため，プロンプトの膨張とコンテキスト長制限に起因する性能劣化が生じるという問題がある。

本研究では，これらの課題を解決するため，複数の LLM エージェントからなるマルチエージェントシステムに基づくサービス合成手法を提案する。各エージェントを特定の API カテゴリや機能領域に特化させることで，プロンプト内

で扱う知識を局所化しつつ，専門性の高いサービス推薦を実現することを目指す。さらに，タスク割当てメカニズムとして契約ネットプロトコルを導入し，マネージャエージェントとコントラクタエージェントがタスク分解と入札・応札を通じて協調的にサービス合成を行う枠組みを設計する。本論文では，マネージャ主導・コントラクタ主導・協調型という三種類のタスク分解プロトコルを比較し，サービス合成精度や単一エージェント方式との性能差を評価することで，LLM ベース・マルチエージェントによるサービス合成の有効性と課題を明らかにする。

そこで本手法の実現にあたり，取り組むべき課題は以下の 2 点である．

タスク分解　自然言語で記述されたユーザ要求から，契約ネットプロトコルに基づいて適切なタスク分解を行う. 要求の粒度や曖昧さに応じて，マネージャ主導・コントラクタ主導・協調型といった異なるタスク分解戦略を切り替えつつも，一貫したサービス合成結果を得られる仕組みが必要となる。

コントラクタエージェントの組織化　エージェントを特定の API カテゴリや機能領域に特化させることでプロンプトの膨張を抑えつつ，サービス仕様や実行環境に関する専門的な知識を十分に反映させる必要がある。また，タスク分解プロトコルの違いがエージェント間の情報共有や推論過程に与える影響を明らかにすることも求められる。

以下，本論文では，第 2 章で複合サービス合成の基本的なアプローチとして水平型・垂直型サービス合成と，LLM を用いたサービス合成について説明する．第 3 章では，契約ネットプロトコルに基づくマルチエージェント方式とタスク分解プロトコルの設計を示す．第 4 章では，評価指標と実験設定を述べ，評価結果を示す．第 5 章で考察を行い，第 6 章で結論と今後の展望を述べる．

第2章　サービス合成

本章では，複合サービスの実現に向けたサービス合成手法について説明する．複合サービスとは，インターネット上に存在する複数の Web サービスを組み合わせて，新たなサービスのことである．近年では，さまざまな Web サービスが API を通じて連携可能となっており，ユーザ要求に応じた柔軟なシステム開発が可能となっている．

複合サービスの代表例として Expedia を挙げる．Expedia は航空券，宿泊，移動手段など複数の API を統合することで，旅行計画・予約を一括提供している．航空券の検索では航空会社情報を提供する API，予約確定時には決済 API，旅の計画閲覧には地図 API を利用し，単独では実現不可能な利便性を提供している．このような複合サービスを開発するためには，ユーザ要求に応じて必要な機能を提供できる Web サービスを適切に発見する必要がある．しかし膨大な数の Web サービスの中から目的に合致したものを探索することは容易ではない．本章では，サービス合成手法について説明する．

2.1　サービス合成手法

従来のサービス合成手法では大きく，水平サービス型と垂直サービス型の 2 つに分類できる．以下でそれぞれについて説明する．

2.1.1　水平型サービス合成

水平型サービス合成とは，あらかじめ与えられたワークフロー（タスク列）に対し，各タスクを実行可能な Web サービス候補の中から，合成全体として望ましい組合せを選択する枠組みである．Zeng らは，同一の機能を提供するサービスが多数存在する状況を想定し，機能差ではなく QoS（非機能的特性） に基づいてサービスを選別する合成方式を提案し [1]．具体的には，価格・応答時間・可用性などの非機能情報から各サービスの QoS ベクトルを定義し，ユーザが与える制約や嗜好（重み付け）を用いて候補を評価する．その上でサービス選択を ローカル最適化 と グローバル最適化 の二通りで扱う．ローカル最適化では，各タスクごとに他タスクとの関係を考慮せず，制約を満たす範囲でスコアが最大となるサービスを選ぶため計算が軽い一方，合成全体の QoS が最適になる保証は弱い．これに対しグローバル最適化では，合成全体の QoS を目的としてタスク間の影響も含めて最適な割当を求めることで，合成結果の品質向上

を狙う．

2.1.2　垂直型サービス合成

垂直型サービス合成は，人工知能のプランニング技術を用いて，ユーザが指定したゴール状態に到達するための Web サービスの実行系列（合成手順）そのものを生成する手法である．Carman らは，サービス合成を計画問題として捉え，オープンで不確実性を含む Web 環境に適した合成アルゴリズムを提案した [2]．提案手法の要点は二つに整理できる．第一に，セマンティック型マッチングである．Web サービス間では，出力と入力のデータ型が厳密に一致しないことが多く，単純なラベル一致に基づく接続判定では合成可能性を過小評価する．そこで，出力型が他サービスの入力型や最終的に達成すべき目標型と意味的に関連するかを，ラベルが指す概念の一般化関係や型構造も含めて判断し，異種スキーマ間の非互換性を緩和する．第二に，探索と実行を統合した逐次的な合成である．サービスの実行結果や入出力値が事前に確定しない状況では，合成計画を最初から完全に構築することが難しいため，候補サービスを選択して実行し，得られた結果を観測して次のステップを更新する手順を繰り返す．このように探索と実行を交互に行うことで，不完全な情報の下でも計画を修正しながら目標達成へ近づく柔軟性を確保する．以上より，本研究は，型の意味的整合性に基づく接続判定と，試行錯誤を許容する逐次実行型の探索戦略を組み合わせ，垂直型サービス合成を実現する先行研究として位置付けられる．

2.2　LLM を用いたサービス合成

LLM を用いたサービス合成では，自然言語で記述されたユーザ要求から，適切な Web サービスを選択・組み合わせる手法である．従来は機能要件を論理式で記述し，形式的なフロー設計が必要であったが，LLM の自然言語理解能力を活用することで，要求の意図や制約を推定し，それに基づいてタスク分割や API 選択を行える．また，API ドキュメントの要約や入力パラメータの生成を LLM が補助することで，ユーザは専門知識なしにサービス合成を実現できる．

一方で，LLM を単純にプロンプト利用するだけでは，API の最新知識不足やハルシネーションにより推薦精度が低下することが指摘されている．さらに BERT 等の従来モデルは API ドメイン固有の知識やマッシュアップの複雑な使用パターンを十分に捉えられず，実運用レベルの推薦に課題が残る．この問題に対し，Qin らは Llama-3.2-3B を基盤とする LLMAR を提案し [3]，指示学習

と多段階ファインチューニングによってAPIドメイン知識をモデル内部に注入する枠組みを示した．また，Matsumotoらは，推論過程を階層的に分解して段階的に進めるGuided Reasoning Chains (GRC) を提案し[4]，要求の抽象度に応じてコア機能抽出，カテゴリ選択，候補列挙，最終推薦という流れで推論を行うようLLMを誘導する．このように推論プロセスを明示化する設計は，要求理解と候補整理を分離する本研究の枠組みとも整合的であり，API推薦の安定性向上に寄与することが示唆される．

LLMARの特徴は，API推薦を単一タスクとして扱わず，以下の4つのサブタスクに分解して学習する点にある．

- API記述生成（AD）：API名から詳細説明を生成し，API機能の理解を深める．
- API分類（AC）：APIのカテゴリを推定し，属性情報を学習する．
- マッシュアップ分類（MC）：要求記述からカテゴリを推定し，構成パターンを獲得する．
- API推薦（AR）：要求記述に対してAPI名を直接生成する．

これらを同時学習するのではなく，AD/AC/MCで基礎知識を注入した後にARへ段階的に特化させることで，タスク間の表現衝突や負の転移を回避し，汎化性能を高めている．

学習効率の面ではLoRAを採用し，巨大モデル全体を更新せずに低ランク行列のみを学習することで，更新パラメータ数を大幅に削減している．実験ではProgrammableWeb由来のAPI・マッシュアップデータで評価され，ベースモデルや大規模汎用LLMを上回る精度が報告された．これらの結果は，LLMをサービス合成に適用する際，ドメイン知識の構造化と多段階学習が有効であることを示している．

第3章　契約ネットプロトコルに基づくサービス合成

本章では，LLM を用いたマルチエージェントシステムに基づくサービス合成手法について説明する．マルチエージェントを用いる中で本研究では契約ネットプロトコル（Contract Net Protocol; CNP）を採用し，エージェント間の協調メカニズムとして利用する．

3.1　契約ネットプロトコル

契約ネットプロトコル（Contract Net Protocol; CNP）は，Reid G. Smith によって提案された分散問題解決のための高水準通信プロトコルであり [5]，分散環境におけるタスク割当てと協調計算を，交渉に基づくメッセージ交換によって実現する枠組みである．CNP では，タスクを外部に委託したい側のエージェントをマネージャ（manager），タスクの実行を引き受ける側のエージェントをコントラクタ（contractor）と呼び，両者の間でタスクの告知，入札，契約締結，結果報告といった一連のやり取りを明示的なプロトコルとして規定している．

典型的な CNP の相互作用は，次のようなステップで構成される．

1. タスク告知（task announcement / call for proposals）
   マネージャは，自身が保持するタスクの内容（目的，前提条件，必要資源，締切，評価基準など）を記述した告知メッセージをブロードキャストし，コントラクタ候補に対して提案を呼びかける．
2. 入札（bidding / proposal submission）
   告知を受け取った各コントラクタは，タスクを遂行可能かどうか，現在の負荷や利用可能資源を考慮して評価する．実行可能であれば，予想コスト，所要時間，達成可能な品質などを含む入札（プロポーザル）をマネージャに送信し，実行困難と判断した場合は辞退メッセージを返す．
3. 契約締結・割当て（award / contract）
   マネージャは，収集した複数の入札を比較し，評価基準に基づいて最も望ましいコントラクタを選定する．選定されたコントラクタには契約成立を示すアワードメッセージを送り，その他のコントラクタには不採択を通知する．これにより，当該タスクに関する一時的なマネージャ–コントラクタ関係が形成される．
4. 実行と結果報告（task execution / status reporting）

コントラクタは割り当てられたタスクを実行し，中間状態や完了結果をステータス報告メッセージとしてマネージャに送信する．マネージャはこれらの報告に基づいてタスクの進捗を把握し，必要であれば再割当てなどの制御を行う．

このように，CNP は単なるタスク割当てアルゴリズムというよりも，タスク告知から結果報告に至るまでのエージェント間通信の流れとメッセージ種別を明確に定義したコミュニケーション・プロトコルとして位置づけられる．各エージェントは，あるタスクに関してはマネージャとして振る舞い，別のタスクに関してはコントラクタとして振る舞うことができるため，システム全体としては固定的な中央管理者を持たない柔軟な階層構造を動的に形成できるという特徴を持つ．また，タスク告知の範囲や入札の条件を調整することで，探索空間の絞り込みや通信量の制御が行える点も利点とされている．本研究では，この CNP をエージェント間の協調メカニズムとして採用し，サービス合成におけるタスク（機能分割や API 選択など）の割当てに応用する．

3.2　システム概要

本研究で提案するサービス合成システムは，LLM を用いたマルチエージェントシステムに基づき，自然言語で与えられた複合サービスの要件から，利用すべき Web サービス API の組み合わせを推薦することを目的としている．

従来の LLM による API 推薦では，要件理解から API 選択までを単一の推

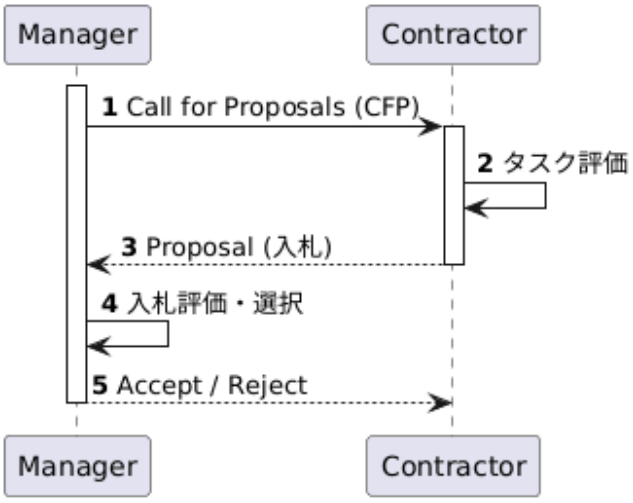

図 1: CNP のシーケンス図

論として一括に処理する設計が多い．しかしこの設計では，多数の API 説明文や仕様を単一プロンプトへ集約する必要があり入力が肥大化しやすい，そして要件や API 群の変動により推論の再現性が低下しやすい．さらに推論過程がブラックボックス化し，誤りが生じた際にどの段階で破綻したかを切り分けにくいといった課題が生じる．各ステップに目的・参照情報・出力形式を与えて推論を誘導する Guided Reasoning Chains の考え方に基づき，サービス合成推論を LLM の内部処理に閉じず，外部化可能な推論タスクの系列として捉える．

本システムは，ユーザ要件を受理しタスクを編成するマネージャエージェント，各 API カテゴリや個別 API に専門化した複数のコントラクタエージェント，および各エージェントが参照する API 説明文から構成される．エージェント間の協調枠組みとして Contract Net Protocol（CNP）を採用し，タスクの公告（Call for Proposals），入札（Proposals），選択（Award）という流れのもとで推論タスクを分担し，最終的な API 推薦結果を得る．CNP を採用する理由は以下の通りである．

1. 専門性に基づく自律的な可否判断を反映するためである．
   本研究では，コントラクタエージェントを API カテゴリまたは個別 API に専門化させ，各エージェントが保持する局所的知識に基づいて「担当可能性」と「候補 API」を判断する設計をとる．CNP は，公告に対して各コントラクタが自律的に入札・辞退を返す手順を持つため，専門性に依存した可否判断を自然に取り込める．
2. 複数候補を比較可能な形で収集し，最終決定を体系化するためである．
   サービス合成では，要求に適合する API 候補が複数存在し得るため，候補の列挙に留まらず，根拠や評価観点を揃えて比較し，最終的な採択を行う必要がある．CNP は複数の入札を収集し，マネージャが採択する流れを規定するため，候補の収集から統合・決定までを一貫した枠組みとして扱える．
3. 責任分担の差分をプロトコルとして明示し，方式間の比較を可能にするためである．
   本研究の焦点は，推論タスクの責任分担（どの推論ステップを誰が担当するか）が，推薦精度に与える影響を明らかにする点にある．CNP は公告・入札・選択という役割とメッセージの流れを明確に定義しているため，責任分担の差分をプロトコルとして記述でき，方式間の比較・検証を体系的

に行える．

その上で，本システムにおけるサービス合成の推論過程を，以下の 4 ステップからなる共通フローとして定義する．

1. コア機能分解：要件文からユーザが最終的に達成したい目的を推定し，必要となる機能単位を抽出する．
2. カテゴリマッピング：抽出した機能を実現し得る API カテゴリ（例：決済，地図，予約管理など）を推定する．
3. API マッチング：推定カテゴリに属する API 説明文と要件に基づいて，適合する候補 API を抽出する．
4. API 選択：抽出した候補 API を比較し，要件適合性の観点から採用 API を確定する．

ここで，(3) は候補 API 集合の抽出（候補提示），(4) は候補集合からの最終採択として区別する．

この 4 ステップは単一のエージェントが一括で実行するのではなく，CNP による公告・入札・選択の枠組みにおいて，マネージャとコントラクタの間で分担される推論タスクとして扱われる．マネージャは要件文の受理，推論タスクの定義と公告，入札結果の集約と整合性確認，および API 群の確定を担う．コントラクタは担当の API 知識に基づき，公告されたタスクに対する判断と候補 API，および根拠を入札として提示する．

本研究では，推論タスクの責任分担（どのステップを誰が担当するか）を CNP 上の設計変数とみなし，マネージャ主導型，コントラクタ主導型，協調型の 3 方式を設計・比較する．この比較により，推論の分割粒度と専門化の度合いが，API 推薦精度および推論コストに与える影響を明らかにする．

3.3 LLM に基づくエージェント

本研究では，自然言語で記述されたユーザ要求および API 説明文を入力として，サービス合成のための API 群を生成する推論主体として，LLM に基づくエージェントを用いる．

本システムにおける LLM エージェントは，マネージャエージェントと複数のコントラクタエージェントから構成される．各エージェントは，CNP に基づく公告・入札・選択の枠組みの中で，自身に割り当てられた推論タスクを実行し，その結果を明示的なメッセージとして外部に出力する．これにより，サー

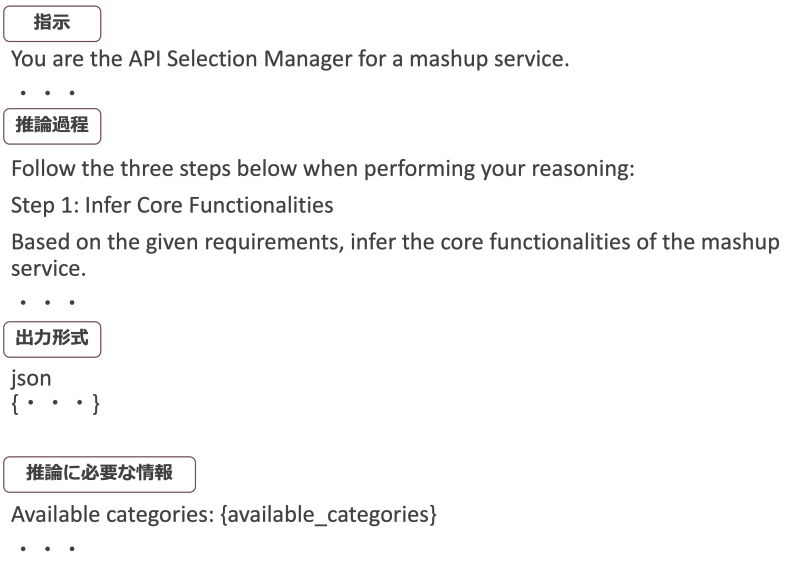

図 2: プロンプト構成の概要

ビス合成推論は LLM の内部推論として閉じるのではなく，外部から観測・比較可能な推論タスクの系列として実現される．

3.3.1 プロンプト設計

本研究では，プロンプトの構成を共通テンプレートとして整理し，各エージェントに同一の順序で提示する．具体的には，(1) 役割や制約などの指示，(2) 推論過程の説明，(3) 出力形式の指定，(4) 推論に必要な情報の提示，という順で記載する．プロンプト構成の概要を図 2 に示す．

3.3.2 コントラクタエージェント

コントラクタエージェントは，個別 API に専門化された LLM エージェントであり，マネージャから公告されたタスク仕様を入力として受け取る．各コントラクタは，自身が参照可能な API 説明文のみを用いて推論を行い，当該タスクを担当可能か否かを判断する．

担当可能と判断した場合，コントラクタエージェントは，適合すると考えられる API 候補と，その判断根拠を入札としてマネージャに返す．ここで提示される API 候補は，システム概要で定義した推論ステップ (3) API マッチングに相当し，候補 API 集合の抽出結果として位置付けられる．一方，担当不可能と判断した場合には，辞退メッセージを返すことで入札を行わない．

3.3.3 マネージャエージェント

マネージャエージェントは，ユーザ要求を受理し，サービス合成全体を統括する役割を担う LLM エージェントである．マネージャは，システム概要で定義した推論過程に基づいて，推論タスクを構成し，CNP に従ってコントラクタエージェントへ公告する．

入札を受信した後，マネージャエージェントは，複数のコントラクタから提示された API 候補と根拠を比較・統合し，要求全体との整合性を確認した上で，最終的に採用する API 集合を確定させる．この処理は，推論ステップ (4) API 選択に対応し，候補集合からの最終採択を意味する．

このように，マネージャエージェントは，分散した推論結果を集約する集中意思決定の役割を担い，コントラクタエージェントは専門性に基づく局所的推論を担うことで，分散推論と統合的意思決定を両立させている．

3.4 プロトコル

本研究では，CNP に基づくサービス合成において，推論タスクの責任分担（どの推論ステップをどのエージェントが担当するか）を設計変数として扱う．具体的には，システム概要で定義した 4 ステップの推論過程（コア機能分解，カテゴリマッピング，API マッチング，API 選択）を，マネージャエージェントとコントラクタエージェントの間でどのように分担させるかによって，異なるサービス合成プロトコルを設計する．

本節では，推論タスクの分担方針が異なるマネージャ主導型，コントラクタ主導型，協調型の 3 方式を定義し，各方式における公告内容，入札内容，および最終的な API 群の確定方法を明確にする．

3.4.1 マネージャ主導型タスク分解

マネージャ主導型では，サービス合成に必要な推論の大部分をマネージャエージェントが担当する．4 ステップの推論過程における役割分担は以下の通りである．

- マネージャの担当：(1) コア機能分解，(2) カテゴリマッピング，(4) API 選択
- コントラクタの担当：(3) API マッチング

本方式では，マネージャエージェントがユーザ要求を解析し，必要なコア機能および対応する API カテゴリを事前に確定した上で，それらをタスク仕様と

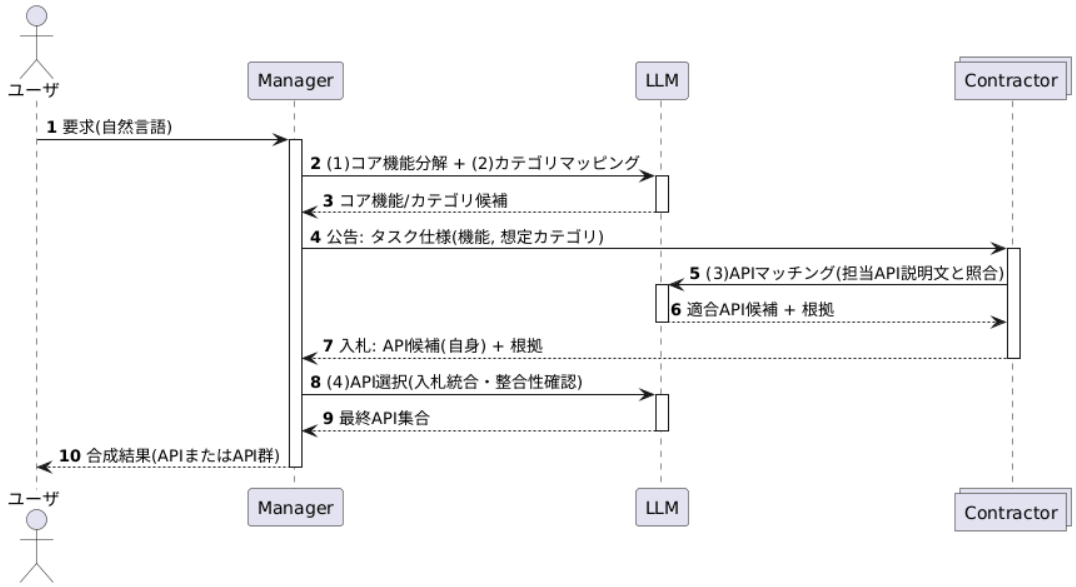

図 3: マネージャ主導プロトコルのシーケンス図

してコントラクタに公告する．公告には，対象となる機能，想定カテゴリが含まれる．

各コントラクタエージェントは，公告された機能及びカテゴリに基づき，自身が参照可能な API 説明文を用いて API マッチングを行い，適合すると判断した API 候補 (自身) とその根拠を入札として返す．マネージャは，収集した入札を比較・統合し，要求全体との整合性を確認した上で，最終的な API 集合を確定させる．図 3 にマネージャ主導プロトコルのシーケンス図を示す．

3.4.2 コントラクタ主導型タスク分解

コントラクタ主導型では，推論の大部分をコントラクタエージェントに委ねる．4 ステップの推論過程における役割分担は以下の通りである．

- マネージャの担当：(4) API 選択
- コントラクタの担当：(1) コア機能分解，(2) カテゴリマッピング，(3) API マッチング

本方式では，マネージャエージェントは，ユーザ要求文そのものをタスクとして公告し，詳細な分解やカテゴリ推定は行わない．各コントラクタエージェントは，自身の担当 API 知識に基づいて要求文を解釈し，コア機能分解およびカテゴリマッピングを行った上で，適合する API 候補と根拠を入札として提示する．

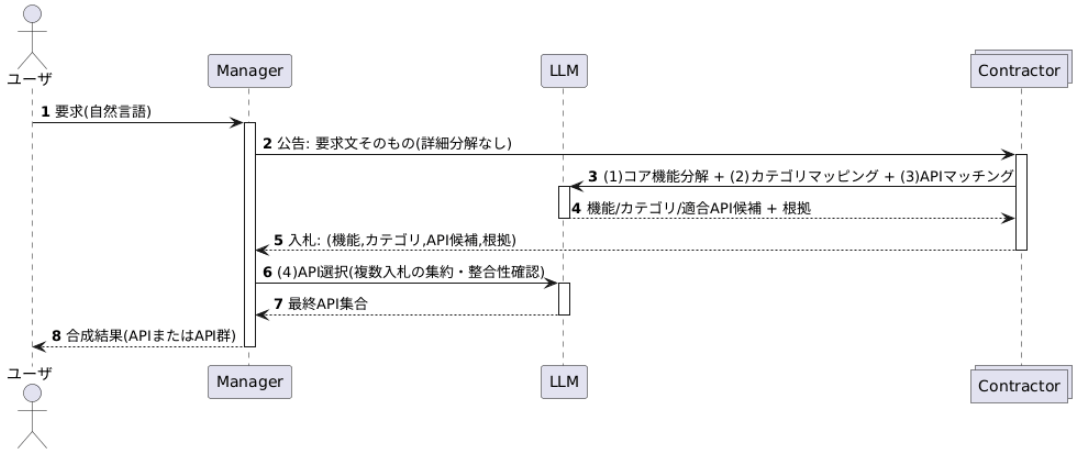

図 4: コントラクタ主導プロトコルのシーケンス図

マネージャエージェントは，複数のコントラクタから提出された入札を集約し，提案内容の整合性を確認した上で，最終的に採用する API 集合を確定させる．図 4 にコントラクタ主導プロトコルのシーケンス図を示す．

3.4.3 協調型タスク分解

協調型では，マネージャとコントラクタが推論タスクを分担し，両者の役割を組み合わせる．4 ステップの推論過程における役割分担は以下の通りである．

- マネージャの担当：(1) コア機能分解
- コントラクタの担当：(2) カテゴリマッピング，(3) API マッチング，(4) API 選択

本方式では，マネージャエージェントがユーザ要求からコア機能分解を行い，抽出した機能単位をタスクとして公告する．各コントラクタエージェントは，公告された機能に対してカテゴリ候補を推定し，担当 API 知識に基づいて API マッチングおよび選択を行い，採用すべき API 候補と根拠を入札として返す．

マネージャは，各入札を統合し，要求全体との整合性を確認した上で，最終的な API 集合を API 群として確定させる．図 5 に協調型プロトコルのシーケンス図を示す．

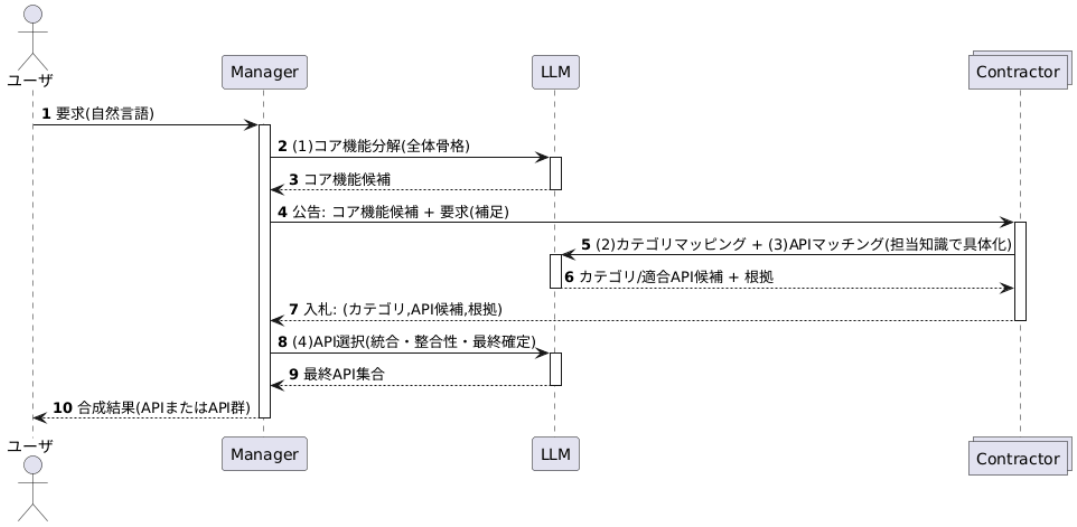

図 5: 協調型プロトコルのシーケンス図

第 4 章　評価

4.1　実験設定

本研究では，提案する CNP ベースのマルチエージェント方式について，推論タスク割り当てプロトコル（マネージャ主導・コントラクタ主導・協調型）を比較する．コントラクタエージェントの組織化については単一 API 単位の設計を採用し，カテゴリ単位で複数 API を担当させる設計は今後の課題とする．いずれの設定においても，ユーザ要求（マッシュアップ要件）を入力として API 群を生成し，正解計画との照合によって計画生成精度を評価する．LLM には gpt-oss20B を用いた．

4.1.1　実験データ

本研究では，サービス推薦モデルを構築するために，ProgrammableWeb に掲載されている複合サービス（Mashup）データおよび原子サービス（Web API）データを使用した．複合サービスデータは，複合サービス名をキーとし，複合サービスカテゴリ，複合サービス説明文，構成サービス（使用 API の一覧），および ID を値として保持する．複合サービスカテゴリおよび構成サービスは複数存在する場合があるため，いずれもリスト型として扱う．原子サービスデータは，原子サービス名をキーとし，サービスカテゴリ，原子サービス説明文，サービス提供者，サービス利用者，および ID を値として保持する．カテゴリ情報は複数付与されており，そのうち先頭要素をメインカテゴリ，それ以外をサブカ

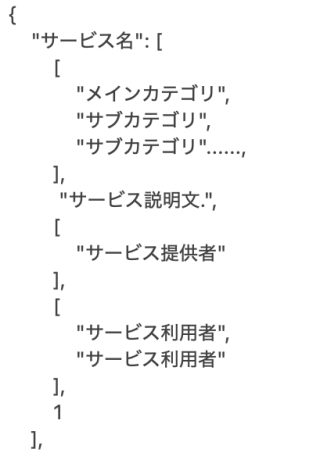

図 6:　原子サービスデータの構成概念

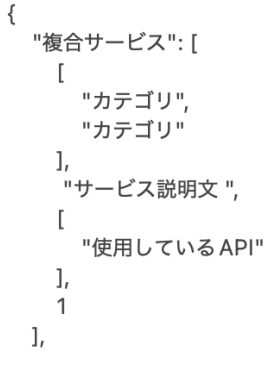

図 7:　複合サービスデータの構成概念

テゴリとして扱う．これらの属性はいずれも ProgrammableWeb 上でユーザにより付与されたものである．複合サービスデータには欠損が含まれるため，前処理としてフィルタリングを行った．フィルタリング条件は，複合サービスが参照する構成サービス（使用 API）が，原子サービスデータに存在しないレコードを除外することである（データ構造は図 2・図 3 に示す）．この処理の結果，複合サービスデータは 4284 件となった．そして原始サービス数は

なお，本節では実データ例ではなく，データ構造の構成概念（スキーマ）を図として示す．原子サービス（Web API）データの構成概念を図 6，複合サービス（Mashup）データの構成概念を図 7 に示す．

また，複合サービスデータには欠損が含まれるため，複合サービスの構成サービス（使用 API）が原子サービスデータに含まれないレコードを除外するフィルタリングを行った．

その中で複合サービス 100 件を抽出し，評価データセットとして用いた．

4.1.2　評価指標

提案手法（CNP に基づくマルチエージェント方式）で生成される API 推薦結果の品質を評価するため，適合率（Precision），再現率（Recall），および F1 スコアを用いる．照合は API 名一致に基づく集合評価として行う．本研究では，最終的な推薦結果だけでなく，プロトコルの各段階における中間結果についても同様の指標を算出し，推論過程のどの段階で精度が変化するかを分析する．具体的には，各プロトコルにおいて以下の段階で評価を行う．

カテゴリ割当て時点　マネージャまたはコントラクタがユーザ要求に対して API カテゴリを推定した段階である．推定されたカテゴリに属する全 API を候補集合 P として，正解集合 G との照合を行う．この段階の評価により，カテゴリレベルでの要求理解の精度を把握できる．

入札時点　コントラクタエージェントが入札として提示した API 候補の集合を P として評価する．この段階では，各コントラクタが自身の担当 API 知識に基づいて選定した候補が含まれるため，カテゴリ割当て時点よりも候補が絞り込まれる．入札時点の評価により，コントラクタの専門的知識がどの程度活用されているかを確認できる．

最終推薦　マネージャエージェントが全コントラクタからの入札を集約・統合し，最終的に採用する API 集合を確定した段階である．この段階の評価が，提案手法全体としての推薦精度を表す．

これら 3 段階の評価を比較することで，各プロトコルにおいて推論のどの段階で精度が向上または低下するかを明らかにし，タスク分解方式の違いが推薦精度に与える影響を詳細に分析する．

加えて，各プロトコルの入出力トークン数を計測し，平均値で比較した．表 1 に示す．

4.1.3　マネージャ主導タスク分解

マネージャ主導方式の結果を示す．本方式では，まずマネージャが要件文をタスク分解し，各タスクに対してカテゴリを割り当てる．その後，要件文とカテゴリを対象カテゴリのコントラクタエージェントに提示して入札を受け，最後に入札結果から適切な API を統合するという流れである．表 2 に示す通り，カテゴリ割当て時点では適合率は 0.111，再現率は 0.607，F1 値は 0.177 となった．入札時点では適合率は 0.047，再現率は 0.485，F1 値は 0.079 となった．適

表 1: 各方式のトークン数（平均）

方式	入力トークン	出力トークン	総トークン
マネージャ主導	9843.45	2716.56	12560.01
コントラクタ主導	1952788.10	999575.98	2952364.07
協調型	522411.36	71828.16	594239.52
単一エージェント	14323.79	28580.34	42904.13

合率は前の時点に比べて低下した．これは，入札で候補 API が拡張され，正解に含まれない候補も多く含まれるためである．最終統合後では適合率は 0.159，再現率は 0.298，F1 値は 0.193 となった．適合率は前の時点に比べて改善した．また，トークン数については，他の方式と比較して最も効率的であった．総トークン数は約 12,560 であり，マネージャが事前にカテゴリを絞り込むことで，不要な通信や処理を抑制できたためと考えられる．

4.1.4　コントラクタ主導タスク分解

コントラクタ主導方式の結果を示す．本方式では，マネージャが要件文そのものをタスクとして公告し，詳細な分解やカテゴリ推定は行わない．表 2 に示す通り，カテゴリ割当て時点では適合率は 0.149，再現率は 0.391，F1 値は 0.193 となった．入札時点では適合率は 0.017，再現率は 0.623，F1 値は 0.031 となり，適合率が大幅に低下した．これは，各コントラクタが独立してカテゴリ推定とマッチングを行うため，全体的な一貫性が欠け，多数の無関係な API が候補として提示されたためと考えられる．最終統合後では適合率は 0.038，再現率は 0.126，F1 値は 0.055 となり，3 方式の中で最も低い性能を示した．コントラクタの局所的な知識を活用する柔軟性は得られるものの，全体的な意見統合の困難さと，各エージェント間の判断基準の不一致が結果の精度低下につながったと考えられる．トークン数は入力約 1,952,788，出力約 999,576，総計約 2,952,364 と 3 方式中最大であり，マネージャによる絞り込みがないまま多数のコントラクタに公告・入札を繰り返した結果，通信量とフィルタリング負荷が爆発的に増大した．

4.1.5　協調型タスク分解

協調型方式の結果を示す．マネージャによるタスク分解とコントラクタによるカテゴリ分解を組み合わせることで，整合性と柔軟性の両立を図る．表 2 に示す通り，カテゴリ割当て時点では適合率は 0.047，再現率は 0.347，F1 値は 0.078 となった．入札時点では適合率は 0.034，再現率は 0.260，F1 値は 0.058 となった．適合率は前の時点に比べて低下した．これは，入札で候補 API が拡張され，正解に含まれない候補も多く含まれるためである．最終統合後では適合率は 0.083，再現率は 0.223，F1 値は 0.114 となった．適合率は前の時点に比べて改善した．また，最終統合後の精度はマネージャ主導方式よりも低く，分担により柔軟性は高まる一方で，トークン数については，3 方式の中で最もコストが高くなった．総トークン数は約 594,240 に達した．これは，マネージャとコ

ントラクタ間での多段階のやり取りが発生し，各ステップでの入出力が増大したためである．提案の統合段階における情報損失が影響していると考えられる．

表 2: 推薦結果の比較

段階	方式	適合率	再現率	F1 値
Step2: カテゴリ割当て時点	マネージャ主導	0.111	0.607	0.177
	コントラクタ主導	0.149	0.391	0.193
	協調型	0.047	0.347	0.078
	単一エージェント	0.174	0.310	0.208
Step3: API マッチング時点	マネージャ主導	0.047	0.485	0.079
	コントラクタ主導	0.017	0.623	0.031
	協調型	0.034	0.260	0.058
	単一エージェント	0.106	0.121	0.105
Step4: API 選択時点	マネージャ主導	0.159	0.298	0.193
	コントラクタ主導	0.038	0.126	0.055
	協調型	0.083	0.223	0.114
	単一エージェント	0.137	0.141	0.130

第5章　考察

5.1　単一エージェントとの比較

単一の汎用 LLM エージェントに多数の API 仕様を集約する方式では，プロンプト肥大化と文脈長制約により推論精度が低下しやすい．一方で，マネージャ主導方式は，マネージャがタスク分解とカテゴリ割当てを行い，コントラクタが担当領域の API 仕様に基づいて入札し，マネージャが整合性を確認して統合するという段階的なプロトコルを持つ．この分担により，単一エージェント方式で生じやすい過剰に広い候補列挙や曖昧要求に対する一括推論の影響が抑えられ，担当領域の文脈を局所化した提案が集約されるため，再現率と適合率の両面で改善が見られた．実際に，単一エージェント方式は適合率 0.137，再現率 0.141 であるのに対し，マネージャ主導方式は適合率 0.159，再現率 0.298 を示しており，タスク分解と入札・統合のプロトコル差が結果の差に寄与していると考えられる．さらに協調型では，適合率 0.083，再現率 0.223 と単一エージェント方式より改善する一方，マネージャ主導方式よりは低い値となった．タスク分解の柔軟性は得られるが，入札結果の統合で情報損失が生じやすいことが影響していると考えられる．カテゴリ割当て時点の F1 値は単一エージェント 0.208 に対しマネージャ主導 0.177 と単一の方が高いが，最終推薦ではマネージャ主導が 0.193 で単一の 0.130 を上回る．これは単一エージェントが初期のカテゴリ分類では良好な一致を示す一方，最終推薦では多数の API を一括処理するため誤候補が混入しやすく，適合率が低下しやすいことによると考えられる．対照的にマネージャ主導は，入札と統合の段階で候補を段階的に精査できるため，初期のカテゴリ推定が多少粗くても最終的な精度を押し上げられる．また，最終推薦に含まれる API 数の平均を比較すると，単一エージェント方式は 5.94 件であり，マネージャ主導（3.19 件）および協調型（3.57 件）よりも多い．これは単一エージェント方式が候補を広く列挙しやすく，過剰に広い候補集合を形成する傾向があることを示している．一方，マルチエージェント方式ではタスク分解とカテゴリ制約によって候補が絞り込まれ，不要な API が最終推薦に残りにくい点が精度面の差に寄与していると考えられる．

表 2 に単一エージェント方式の結果も示す．

5.2　プロトコル間の比較考察

本研究で比較した 3 つのプロトコル（マネージャ主導型，コントラクタ主導型，協調型）について，推薦精度，推論コスト，およびシステムの振る舞いの観点から詳細な考察を行う．

5.2.1　推薦精度の観点

推薦精度の比較において，マネージャ主導型が最もバランスの取れた性能（適合率 0.159，再現率 0.298）を示した．この要因として，マネージャによる事前のカテゴリ絞り込み（トップダウンアプローチ）が，探索空間を適切に限定し，無関係な API からのノイズ（誤った入札）を効果的に排除できた点が挙げられる．協調型（適合率 0.083，再現率 0.223）は，コントラクタ側でのカテゴリ推定とマネージャ側での統合という柔軟な手続きを持つものの，マネージャ主導型ほどの精度には達しなかった．これは，各コントラクタが局所的な視点でカテゴリ適合性を判断するため，システム全体としての一貫性が保ちにくく，結果としてマネージャが統合時に正解候補を見落とす，あるいは誤った候補を採用する頻度が高まったためと推察される．コントラクタ主導型（適合率 0.038，再現率 0.126）は，3 方式の中で最も低い精度を示した．全コントラクタが独立して要件解釈とカテゴリ推定を行うため，判断の発散が極めて大きく，無関連な API が過剰に列挙される結果となった．各エージェント間の判断基準が統一されず，統合段階で情報損失が大きく発生したためと考えられる．以上の結果は，LLM を用いたマルチエージェントシステムにおいて，完全な自律分散（ボトムアップ）よりも，ある程度の中央制御（トップダウンの構造化）を残した階層的な意思決定の方が，サービス合成のような複雑なタスクにおいては有効であることを示唆している．

5.2.2　推論コストと効率性の観点

推論コスト（トークン数）の観点では，マネージャ主導型の優位性が顕著であった．その総トークン数（約 1.2 万）は，協調型（約 59 万）の約 50 分の 1，単一エージェント（約 5 万）と比較しても約 4 分の 1 に留まった．これは，マネージャ主導型が初期段階でタスクと対象カテゴリを明確に定義し，関係するコントラクタ（API）のみに入札を求めるという「選択的な通信」を実現しているためである．コントラクタ主導型でも同様の傾向が予想されたが，各コントラクタが要件を独立解釈するため，実際には大量の関連エージェントへの問

表 3: 最終推薦 API 数（平均）

方式	平均 API 数
マネージャ主導	3.19
協調型	3.57
コントラクタ主導	4.61
単一エージェント	5.94

い合わせと，返ってきた大量の無関係な候補処理が必要になり，通信量とフィルタリング負荷が増大した．実測値として，コントラクタ主導型の総トークン数は約 2.95 百万（入力約 1.95 百万，出力約 1.00 百万）に達し，マネージャ主導型の約 236 倍，協調型の約 5 倍に相当する．対照的に，協調型では，マネージャとコントラクタ間での多段階のネゴシエーションが発生し，各ステップでの入出力が増大した．特に協調型では，柔軟性を担保するための調整コストが非常に高くつくことが明らかになった．実運用を想定した場合，API 数の増加に対するスケーラビリティの観点からも，通信量を抑制できるマネージャ主導型の設計が最も合理的であると言える．

5.2.3 最終推薦 API 数の観点

最終推薦に含まれる API 数の平均は表 3 に示すように，マネージャ主導 3.19 件，協調型 3.57 件，コントラクタ主導 4.61 件，単一エージェント 5.94 件であった．単一エージェントは候補を広く列挙しやすく，過剰な候補集合を形成する傾向が見られる．一方でマルチエージェント方式では，タスク分解とカテゴリ制約により候補が整理され，不要な API が最終推薦に残りにくいことが，適合率や F1 の改善に寄与していると考えられる．

5.2.4 責任分担と柔軟性のトレードオフ

各プロトコルの特性を責任分担の視点から整理すると，以下のトレードオフが確認できる．マネージャ主導型はマネージャの負荷・責任が大きく，マネージャが初期のカテゴリ判定を誤ると，その後のマッチングで挽回することが難しいというリスク（単一障害点に近い特性）を持つ．しかし，今回の実験結果からは，そのリスクよりも探索効率向上のメリットが上回ることが示された．協調型やコントラクタ主導型はコントラクタの自律性を重視し，マネージャの判断ミスを局所的な知識で補完できる可能性を持つが，その代償として統合の複雑

さと通信コストが増大する．特にコントラクタ主導型では，各エージェントが独立した判断基準を持つため，統合段階での意見調整が極めて困難になり，結果として最も低い精度に陥ったと考えられる．今回の実験設定においては，効率性を重視したマネージャ主導型が有利に働いたが，要件が極めて曖昧でカテゴリ定義自体が困難なケースや，未知の API を探索的に発見する必要があるケースにおいては，協調型のような柔軟なアプローチが再評価される可能性も残されている．

5.3 今後の課題

今後の課題として，まず gpt-oss20B 以外のモデルで同一条件の比較実験を行い，モデル依存性や頑健性を検証する必要がある．モデルのサイズや学習方針の違いが，タスク分解の安定性や入札専門 API 参照により局所知識が効いた可能性がある．再現率が単一 0.141 に対しマネージャ 0.298、協調 0.223 と大きく改善の一貫性にどの程度影響するかを明らかにすることで，手法の一般性を高めることができる．次に，現状はコントラクタエージェントに 1 API のみを割り当てているため，1 エージェントに複数 API を持たせた場合の推薦精度や推論コストの変化を評価し，専門性と分担効率のトレードオフを明らかにする必要がある．さらに，複数 API を担当させた場合にカテゴリ境界が曖昧になる可能性や，入札段階での候補過剰生成が起きるかどうかを確認し，適切な割当粒度を検討する．加えて，コントラクタ主導方式は各コントラクタが API 要件を直接参照できるため，担当 API 数を増やすことでタスク分解の妥当性や入札の質が向上する可能性がある．今後は，担当 API 数を段階的に増やした比較実験を行い，コントラクタ主導方式の性能改善とその条件を検証する．

第6章　おわりに

本論文では，LLM を用いたマルチエージェントシステムに基づく Web サービス合成手法を提案した．契約ネットプロトコルに基づくタスク分解メカニズムを導入し，マネージャ主導・コントラクタ主導・協調型という三種類のタスク分解プロトコルを設計・比較した．また，入札エージェントの組織化としては，単一 API 単位の割当てを採用し，カテゴリ単位で複数 API をまとめて担当させる設計は今後の課題とした．

本研究の貢献は以下である．本研究で設定した課題に対して，次のように具体的な検証と知見を示した．

推論タスク割り当て　推論プロセスを CNP に基づく責任分担モデルとして形式化し，マネージャ主導・コントラクタ主導・協調型の 3 方式を比較した．マネージャ主導ではトップダウンでカテゴリを絞り込み，探索空間とノイズを抑えつつ高い適合率・再現率を達成した．協調型では柔軟性はあるものの，通信コストと統合時の情報損失が課題となった．要求の粒度や曖昧さに応じてプロトコルを切り替える適応設計の必要性を示した．

コントラクタエージェントの組織化　API 知識を単一 LLM に集約するのではなく，コントラクタに分散保持させることで，コンテキスト長の制約とハルシネーションを軽減できることを示した．1 API 単位で専門化した構成でも，大規模な API 群に対するスケーラビリティと更新耐性を確保できることを確認した．カテゴリ単位で複数 API を担当させる粒度最適化は，今後の検証課題として残る．

今後の展望として，以下の課題に取り組むことで，より実用的で柔軟なサービス合成システムの実現を目指す．

エージェントの組織化では，1 エージェント 1 API から，類似 API を束ねたカテゴリ担当エージェントへ拡張した際の精度・通信量・推論負荷を評価し，最適な粒度を求める．基盤モデルについては，大規模モデルと軽量モデルの双方で同一プロトコルを検証し，モデル特性が自律的判断や統合精度に与える影響を測定して頑健性と汎用性を高める．

謝辞

本研究を行うにあたり, 熱心なご指導, ご助言を賜りました指導教官の村上陽平教授を並びに松本賢司さんに深く感謝を申し上げます.

参考文献

[1] Zeng, L., Benatallah, B., Ngu, A. H. H., Dumas, M., Kalagnanam, J. and Chang, H.: QoS-aware middleware for Web services composition, *IEEE Transactions on Software Engineering*, Vol. 30, No. 5, pp. 311–327 (2004).

[2] Carman, M. A., Serafini, L. and Traverso, P.: Web Service Composition as Planning, *ICAPS 2003 Workshop on Planning for Web Services*, pp. 1636–1642 (2003).

[3] Qin, S., Zhao, Y., Wu, H., Zhang, L. and He, Q.: Harnessing the Power of Large Language Model for Effective Web API Recommendation, *IEEE Transactions on Industrial Informatics*, Vol. 21, No. 7 (2025).

[4] Matsumoto, K.: Guided Reasoning Chains for API Recommendation, *Proceedings of International Conference on Service-Oriented Computing (ICSOC)* (2025).

[5] Smith, R. G.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver, *IEEE Transactions on Computers*, Vol. C-29, No. 12 (1980).