(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2020年2月13日(13.02.2020)

WIPO | PCT

(10) 国際公開番号
WO 2020/031845 A1

(51) 国際特許分類:
G06F 8/73 (2018.01)

(21) 国際出願番号： PCT/JP2019/030228

(22) 国際出願日： 2019年8月1日(01.08.2019)

(25) 国際出願の言語： 日本語

(26) 国際公開の言語： 日本語

(30) 優先権データ：
特願 2018-148615 2018年8月7日(07.08.2018) JP

(71) 出願人: 日本電信電話株式会社 (**NIPPON TELEGRAPH AND TELEPHONE CORPORATION**) [JP/JP]; 〒1008116 東京都千代田区大手町一丁目５番１号 Tokyo (JP).

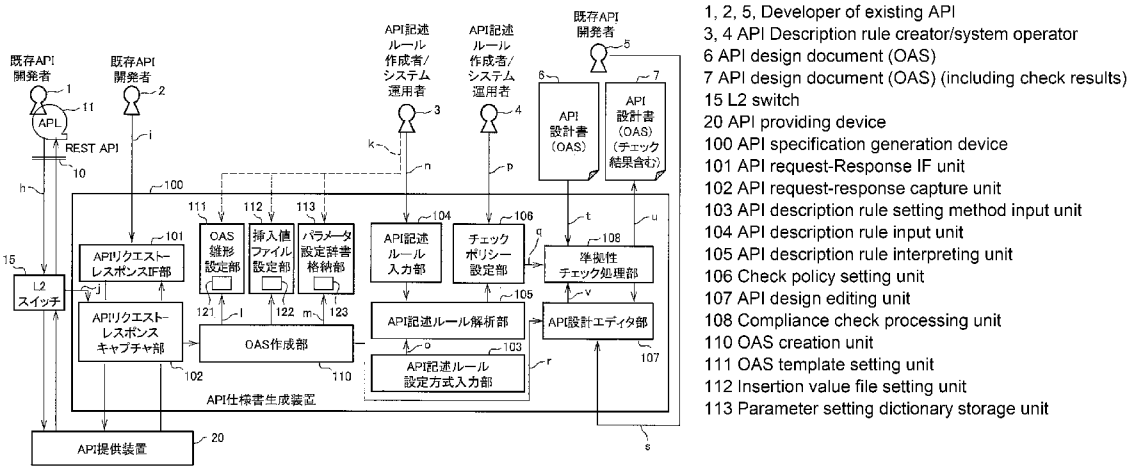
(72) 発明者: 小俣 真吾(**OMATA, Shingo**); 〒1808585 東京都武蔵野市緑町３丁目９－１１ ＮＴＴ知的財産センタ内 Tokyo (JP).

(74) 代理人: 特許業務法人磯野国際特許商標事務所(**ISONO INTERNATIONAL PATENT OFFICE, P.C.**); 〒1050001 東京都港区虎ノ門一丁目１番１８号 ヒューリック虎ノ門ビル Tokyo (JP).

(81) 指定国(表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ,

(54) **Title:** API SPECIFICATION GENERATION DEVICE, API SPECIFICATION GENERATION METHOD, AND PROGRAM

(54) 発明の名称： ＡＰＩ仕様書生成装置、ＡＰＩ仕様書生成方法、およびプログラム



1, 2, 5, Developer of existing API
3, 4 API Description rule creator/system operator
6 API design document (OAS)
7 API design document (OAS) (including check results)
15 L2 switch
20 API providing device
100 API specification generation device
101 API request-Response IF unit
102 API request-response capture unit
103 API description rule setting method input unit
104 API description rule input unit
105 API description rule interpreting unit
106 Check policy setting unit
107 API design editing unit
108 Compliance check processing unit
110 OAS creation unit
111 OAS template setting unit
112 Insertion value file setting unit
113 Parameter setting dictionary storage unit

(57) **Abstract:** The present invention provides an API specification generation device, an API specification generation method, and a program, which are capable of generating an OAS specification at low cost on the basis of an API request-response. This API specification generation device 100 comprises: an OAS template setting unit 111 which sets in advance a template file 121 for creating an OAS specification 127; an insertion value file setting unit 112 which sets an insertion value file 122 describing parameter setting logic used in the OAS specification 127; an API request-response capture unit 102 which captures an API request and a response thereto; and an OAS creation unit 110 which sets parameters by reflecting the captured API request-response data and the reference result of the insertion value file 122 to the template file 121, and creates the OAS specification 127 on the basis of the set parameters.

WO 2020/031845 A1

［続葉有］

NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) 指定国(表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

添付公開書類:

一 国際調査報告(条約第21条(3))

(57) 要約:ＡＰＩリクエストーレスポンスをもとに低コストでＯＡＳ仕様書を作成できるＡＰＩ仕様書生成装置、ＡＰＩ仕様書生成方法、およびプログラムを提供する。ＡＰＩ仕様書生成装置１００は、ＯＡＳ仕様書１２７を作成する際の雛形ファイル１２１をあらかじめ設定するＯＡＳ雛形設定部１１１と、ＯＡＳ仕様書１２７で用いるパラメータ設定ロジックを記述する挿入値ファイル１２２を設定する挿入値ファイル設定部１１２と、ＡＰＩリクエストとそのレスポンスをキャプチャするＡＰＩリクエストーレスポンスキャプチャ部１０２と、キャプチャしたＡＰＩリクエストーレスポンスデータと、挿入値ファイル１２２の参照結果とを雛形ファイル１２１に反映してパラメータを設定し、設定したパラメータをもとにＯＡＳ仕様書１２７を作成するＯＡＳ作成部１１０と、を備える。

明　細　書

発明の名称 ：

ＡＰＩ仕様書生成装置、ＡＰＩ仕様書生成方法、およびプログラム

技術分野

[0001] 本発明は、ＡＰＩ仕様書生成装置、ＡＰＩ仕様書生成方法、およびプログラムに関する。

背景技術

[0002] 各事業者がサービスを管理するためのＡＰＩ（Application Program Interface）をウェブ上にて提供しており、ユーザはＡＰＩを呼び出すソフトウェアを通じてサービスを構築・運用している。ＡＰＩの仕様（ＡＰＩの名称、ＵＲＬ（Uniform Resource Locator）、パラメータ等）は、各事業者が規定しており、ユーザは当該規定に従ってＡＰＩを使用する。

[0003] 既存機能やＡＰＩ仕様をＯＡＳ（Open API spec）仕様に変換する技術として、非特許文献１，２に記載された技術がある。また、ＯＡＳを活用した関連技術には、非特許文献３～５に記載された技術がある。

[0004] 非特許文献１には、データベース（ＤＢ）からWeb APIをドキュメント(Open API標準)と共に自動作成するツールが記載されている。

[0005] 非特許文献２には、ＡＰＩリクエストーレスポンスから簡易的なＯＡＳ仕様書を自動作成するツールが記載されている。

[0006] 非特許文献３には、存在従属グラフからSwagger仕様書の生成クラス図のようなドメインモデルからSwagger Specを作成する技術が記載されている。

[0007] 非特許文献４には、ＡＰＩ設計書に記載されているＡＰＩ特有のパラメータ（ＡＰＩ概要、メソッド等）を解析し、Semantic的エラーを検出する技術が記載されている。

[0008] 非特許文献５には、開発済のＡＰＩを、ＴＭＦ（TM Forum members）提供のＣＴＫ（Conformance Test Kits）に従いNewmanというツールを用いてＡＰＩリクエストを送ると、Syntax的にＴＭＦ準拠になっているかチェックする

ツールが記載されている。このツールは、テスト結果をhtmlやjsonファイルとして生成する。

[0009] ＯＡＳベースで設計／（「／」は、「および／または」、を表記する。以下同様。）開発されていない既存ＡＰＩ（以降、「既存ＡＰＩ」という)仕様を標準仕様書であるＯＡＳ形式にする場合について説明する。

図１２は、ＯＡＳ仕様書作成を説明する図であり、（ａ）はその全体構成図、（ｂ）はその動作説明図である。図１２（ａ）中、二重線で示される表記は、ＡＰＩを示す（以下、各図において同様）。図１３は、図１２の簡易ＯＡＳフォーマット作成ツールで用いるＡＰＩリクエストおよびＡＰＩレスポンスの一例を示す図である。

図１２（ａ）に示すように、ＡＰＬ（Application）サービス（以下、既存ＡＰＬという）１１と既存機能１２（既存機能Ａ１２という）との間には、インターフェースであるＲＥＳＴ（REpresentational State Transfer） ＡＰＩ１０が設けられている。なお、ＲＥＳＴ ＡＰＩ１０は、ＨＴＴＰ以外のプロトコルに対応した機能はない。

[0010] 既存ＡＰＩからＯＡＳ仕様書を作成する場合、簡易ＯＡＳフォーマット作成ツール（図１２（ａ）の符号ａ参照）を用いて、既存ＡＰＩのリクエスト－レスポンスを解析する。例えば、図１２（ｂ）に示すＡＰＩリクエスト－レスポンス１２６（ＨＴＴＰ）を解析する。ＡＰＩリクエスト－レスポンス１２６は、図１３に示すＡＰＩリクエスト１２４およびＡＰＩレスポンス１２５からなる。

[0011] 図１２（ｂ）の符号ｂに示すように、簡易ＯＡＳフォーマット作成ツールを用いて、機能ＡについてＯＡＳ形式のフォーマット（機能Ａ ＯＡＳフォーマット２０）を生成する。

図１２（ｂ）の符号ｃに示すように、開発者が、生成された機能Ａ ＯＡＳフォーマット２０に対してパラメータ等の修正を加えて、機能Ａ ＯＡＳ仕様書２１を作成する。

機能Ａ ＯＡＳフォーマット２０に、微修正を加えて、機能Ａ ＯＡＳ仕様

書２１を作成している。

先行技術文献

非特許文献

[0012] 非特許文献1：API Server: データベースからREST API を超高速で自動生成, ［online］,［平成３０年７月１５日検索］,インターネット 〈URL: https://www.cdata.com/jp/apiserver/〉

非特許文献2：Swagger inspector, ［online］,［平成３０年７月１５日検索］,インターネット 〈URL: https://inspector.swagger.io/builder〉

非特許文献3：井田明男, 他２名, 「存在従属グラフからSWAGGER仕様書の生成」, 信学技報, IEICE Technical Report KBSE2016-29(2016-11)

非特許文献4：小俣真吾, 「API記述ルールに準拠したAPI設計支援方法に関する一検討」, 第14 回NWS 研究会, Oct., 2017

非特許文献5：CTK(Compatibility Test Kit), ［online］,［平成３０年７月１５日検索］,インターネット 〈URL: github.com/tmforum/CustomerManagementCTK〉

発明の概要

発明が解決しようとする課題

[0013] 上述したように、既存のＲＥＳＴ ＡＰＩのＡＰＩリクエストーレスポンス１２６の解析のみで簡易なＯＡＳ仕様書２１の作成は可能である。

しかしながら、（１）単純なＡＰＩリクエストーレスポンス１２６のみの解析では、ＯＡＳの仕様を決められないパラメータ（データ型、データ範囲、値の最大値／最小値等）が存在する。例えば、データ型（“string”, “number”, “integer”, “boolean”, “array”等）の決定や値の範囲（maximum minimum）を一意に特定してＯＡＳに反映させるのは難しく、加筆修正する必要がある。そのため、ＯＡＳ仕様書を自動生成させても、ＯＡＳ仕様書を修正する稼働が発生する。また、ＨＴＴＰ（Hypertext Transfer Protocol）以外のプロトコルに対応したＯＡＳ仕様書の自動生成技術は存在しない

。

[0014] （２）ＡＰＩリクエスト－レスポンス１２６のデータから、どのようなロジックでＯＡＳパラメータに反映するかのルールを設定する機能は存在しない。このため、各社独自パラメータやＨＴＴＰ以外のプロトコルに対応できていない。

[0015] 以上の理由から、非特許文献１～５に記載された技術ついても下記の課題がある。

非特許文献１に記載の技術は、特定のＤＢから機械的にパラメータを設定可能な領域に限定した簡易なＯＡＳを作成することはできるものの、一意に特定不可なパラメータまでは設定できない。また、パラメータ設定のロジックを組むことも不可である。

[0016] 非特許文献２に記載の技術は、パラメータを機械的に、設定可能な領域に限定した簡易なＯＡＳを作成することはできるものの、一意に特定不可なパラメータ（データ型、値の範囲等）までは設定できない。また、パラメータ設定のロジックを組むことも不可である。さらに、ＨＴＴＰのみに対応であり、他のプロトコルに対応していない。

[0017] 非特許文献３に記載の技術は、ＡＰＩリクエスト－レスポンスの結果をもとに、Swagger Specを作成する技術については触れられていない。

[0018] 非特許文献４に記載の技術は、ＯＡＳ仕様書の作成について手書きを前提としており、既存ＡＰＩのリクエストレスポンスを解析してＯＡＳ仕様書を自動生成する機能には触れられていない。また、非特許文献４に記載の技術は、カスタマイズ領域が特定の記述ルールに準拠しているか否かをチェックするチェックルールに限定されている。非特許文献４に記載の技術は、ＡＰＩリクエスト－レスポンスパラメータの値をもとにＯＡＳ仕様書を作成する機能については触れられていない。

[0019] 非特許文献５に記載の技術は、特定の記述ルールのうちSyntaxエラーを検出するツールであり、非特許文献５に記載の技術は、特定のルールへの準拠性を確認するのみである。このため、非特許文献５に記載の技術では、準拠

性を確認するルールのカスタマイズは難しい。また、開発済ＡＰＩのリクエストーレスポンスの結果からSwagger Specを自動生成する技術は具備していない。

[0020] 上記非特許文献１～５のいずれの技術についても、ＡＰＩリクエストーレスポンスからデータ型や値の範囲まで反映するＯＡＳ仕様書の自動生成機能は提案されていない。様々なプロトコルに対応可能なカスタマイズ性を担保した機能も述べられていない。

[0021] このような背景に鑑みて本発明がなされたのであり、本発明は、ＡＰＩリクエストーレスポンスをもとに低コストでＲＥＳＴ ＡＰＩの標準仕様書であるＯＡＳ仕様書を作成できるＡＰＩ仕様書生成装置、ＡＰＩ仕様書生成方法、およびプログラムを提供することを課題とする。

課題を解決するための手段

[0022] 前記した課題を解決するため、請求項１に記載の発明は、既存ＡＰＩの仕様を変換して標準仕様のＯＡＳ仕様書を生成するＡＰＩ仕様書生成装置であって、前記ＯＡＳ仕様書を作成する際の雛形ファイルをあらかじめ設定するＯＡＳ雛形設定部と、前記ＯＡＳ仕様書で用いるパラメータ設定ロジックを記述する挿入値ファイルを設定する挿入値ファイル設定部と、ＡＰＩリクエストとそのレスポンスをキャプチャするＡＰＩリクエストーレスポンスキャプチャ部と、キャプチャしたＡＰＩリクエストーレスポンスデータと、前記挿入値ファイルの参照結果とを前記雛形ファイルに反映してパラメータを設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成するＯＡＳ作成部と、を備えることを特徴とするＡＰＩ仕様書生成装置とした。

[0023] また、請求項４に記載の発明は、既存ＡＰＩの仕様を変換して標準仕様のＯＡＳ仕様書を生成するＡＰＩ仕様書生成装置が、前記ＯＡＳ仕様書を作成する際の雛形ファイルをあらかじめ設定するステップと、前記ＯＡＳ仕様書で用いるパラメータ設定ロジックを記述する挿入値ファイルを設定するステップと、ＡＰＩリクエストとそのレスポンスをキャプチャするステップと、キャプチャしたＡＰＩリクエストーレスポンスデータと、前記挿入値ファイ

ルの参照結果とを前記雛形ファイルに反映してパラメータを設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成するステップと、を実行することを特徴とするＡＰＩ仕様書生成方法とした。

[0024] また、請求項５に記載の発明は、既存ＡＰＩの仕様を変換して標準仕様のＯＡＳ仕様書を生成するＡＰＩ仕様書生成装置としてのコンピュータを、前記ＯＡＳ仕様書を作成する際の雛形ファイルをあらかじめ設定するＯＡＳ雛形設定手段、前記ＯＡＳ仕様書で用いるパラメータ設定ロジックを記述する挿入値ファイルを設定する挿入値ファイル設定手段、ＡＰＩリクエストとそのレスポンスをキャプチャするＡＰＩリクエスト－レスポンスキャプチャ手段、キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記挿入値ファイルの参照結果とを前記雛形ファイルに反映してパラメータを設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成するＯＡＳ作成手段、として機能させるためのプログラムとした。

[0025] このようにすることで、ＡＰＩリクエスト－レスポンスをもとに低コストでデータ型や値の範囲まで反映するＯＡＳ仕様書を作成できる。また、各種プロトコルの値から、ＯＡＳにパラメータをどのように設定するかを記述するパラメータ設定ロジックも、システム運用者にてカスタマイズ可能である。さらに、独自パラメータやＨＴＴＰ以外のプロトコルもＯＡＳ仕様書の対象にすることができる。

[0026] また、請求項２に記載の発明は、一意にＯＡＳ仕様書へパラメータを登録するパラメータ設定辞書を格納するパラメータ設定辞書格納部をさらに備え、前記ＯＡＳ作成部は、キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記挿入値ファイルの参照結果とを前記雛形ファイルに反映して暫定パラメータを設定するとともに、キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記パラメータ設定辞書を参照して、前記暫定パラメータを、当該暫定パラメータよりも精度の高いパラメータに設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成することを特徴とする請求項１に記載のＡＰＩ仕様書生成装置とした。

[0027] このようにすることで、ＡＰＩリクエスト－レスポンスとＡＰＩパラメータ設定辞書を参照して、一意に特定不可であるパラメータも高精度で設計可能となる。このため、ＯＡＳ仕様書の作成および修正稼働を削減することができる。

[0028] また、請求項３に記載の発明は、アプリケーションプログラミングインタフェースを設計する際のルールを記述したＡＰＩ記述ルールを入力するＡＰＩ記述ルール入力部と、前記ＡＰＩ記述ルールを解析してチェックポリシーを作成するＡＰＩ記述ルール解析部と、前記チェックポリシーに基づいて前記ＡＰＩ記述ルールに準拠していることをチェックする準拠性チェック処理部と、をさらに備え、前記ＯＡＳ作成部は、作成した前記ＯＡＳ仕様書を前記準拠性チェック処理部に送り、前記準拠性チェック処理部は、前記ＯＡＳ作成部から送られた前記ＯＡＳ仕様書をもとに、前記ＡＰＩ記述ルールへの準拠性をチェックすることを特徴とする請求項１に記載のＡＰＩ仕様書生成装置とした。

[0029] このようにすることで、生成されたＡＰＩ仕様書について、特定のＡＰＩ記述ルールへの準拠性も確認することができる。

発明の効果

[0030] 本発明によれば、ＡＰＩリクエスト－レスポンスをもとに低コストでＯＡＳ仕様書を作成できるＡＰＩ仕様書生成装置、ＡＰＩ仕様書生成方法、およびプログラムを提供することができる。

図面の簡単な説明

[0031] [図1]本発明の実施形態に係るＡＰＩ仕様書生成装置の概要を示す図であり、（ａ）はその全体構成図、（ｂ）はその動作説明図である。

[図2]本発明の実施形態に係るＡＰＩ仕様書生成装置を示す構成図である。

[図3]本発明の実施形態に係るＡＰＩ仕様書生成装置のＯＡＳ雛形設定部が設定する雛形ファイルの一例を示す図である。

[図4]本発明の実施形態に係るＡＰＩ仕様書生成装置の挿入値ファイル設定部が設定する挿入値ファイルの一例を示す図である。

[図5]本発明の実施形態に係るＡＰＩ仕様書生成装置のＯＡＳ作成部の動作を示す図である。

[図6]本発明の実施形態に係るＡＰＩ仕様書生成装置のＯＡＳ作成部の動作により作成されたＯＡＳ仕様書を示す図である。

[図7]本発明の実施形態に係るＡＰＩ仕様書生成装置のＳＩＰをＨＴＴＰ形式のＲＥＳＴ ＡＰＩに変換する場合、ＯＡＳ雛形設定部が設定する雛形ファイルの一例を示す図である。

[図8]本発明の実施形態に係るＡＰＩ仕様書生成装置の挿入値ファイル設定部が設定する挿入値ファイルの一例を示す図である。

[図9]本発明の実施形態に係るＡＰＩ仕様書生成装置が規定する注釈様式に従って、構文的記述ルールを記載した例を示す図である。

[図10]本発明の実施形態に係るＡＰＩ仕様書生成装置のＡＰＩ記述ルールを表にして示す図である。

[図11]本発明の実施形態に係るＡＰＩ仕様書生成装置が規定する記述様式に従って、メソッド使用ルールを記載した例を示す図である。

[図12]従来技術のＯＡＳ仕様書作成を説明する図であり、（ａ）は全体構成図、（ｂ）は動作説明図である。

[図13]図１２の簡易ＯＡＳフォーマット作成ツールで用いるＡＰＩリクエストおよびＡＰＩレスポンスの一例を示す図である。

発明を実施するための形態

[0032] 以下、図面を参照して本発明を実施するための形態（以下、「本実施形態」という）におけるＡＰＩ仕様書生成装置等について説明する。

（原理説明）

図１は、本発明の実施形態に係るＡＰＩ仕様書生成装置の概要を示す図であり、（ａ）はその全体構成図、（ｂ）はその動作説明図である。図１２および図１３と同一構成部分には同一符号を付している。

ＡＰＩ仕様書生成装置１００は、既存ＡＰＩの仕様を自動で標準仕様の仕様書に変換し生成する。

図１（ａ）に示すように、ＡＰＩ仕様書生成装置１００は、ＲＥＳＴ ＡＰＩ１０と既存機能１２間で送受信されるＡＰＩリクエストーレスポンスをキャプチャ可能に設置される（詳細後記）。

図１（ｂ）の符号ｄに示すように、キャプチャブロック＃１でキャプチャされたＡＰＩリクエストーレスポンス１２６は、ＯＡＳ生成処理ブロック＃２に出力される。

[0033] ＯＡＳ生成処理ブロック＃２は、ＯＡＳ雛形設定部１１１（後記）、挿入値ファイル設定部１１２（後記）、パラメータ設定辞書１２３（後記）を有する。

ＯＡＳ生成処理ブロック＃２では、パラメータ設定ロジックを含むＡＰＩガイドラインファイルに従い、あらかじめ用意してある雛形ファイル１２１に挿入値ファイル１２２の参照結果とパラメータを反映する（図１（ｂ）の符号ｅ参照）。

また、一意に特定不可であるパラメータについては、例えばＡＰＩガイドラインファイルのパラメータ設定ロジックに従い、パラメータ設定辞書１２３を参照の上、パラメータを設定する（図１（ｂ）の符号ｆ参照）。

[0034] ＡＰＩ仕様書生成装置１００は、特定のＡＰＩ記述ルールへの準拠性も確認する。すなわち、図１（ｂ）の符号ｇに示すように、ＯＡＳ生成処理ブロック＃２で生成したＯＡＳをＯＡＳ出力ブロック＃３に出力し、特定のＡＰＩ記述ルールへの準拠性も確認する（後記）。

[0035]（実施形態）

図２は、本発明の実施形態に係るＡＰＩ仕様書生成装置を示す構成図である。

図２に示すように、既存ＡＰＬ１１は、ＲＥＳＴ ＡＰＩ１０およびＬ２（layer2）スイッチ１５を介してＡＰＩ提供装置（既存機能）２０に接続される。既存ＡＰＬ１１は、既存ＡＰＩ開発者１が使用するものとする。Ｌ２スイッチ１５は、パケットのＭＡＣ（Media Access Control）アドレスにより中継先を判断して中継動作を行う。

[0036] ＡＰＩ仕様書生成装置１００は、ＲＥＳＴ ＡＰＩ１０とＡＰＩ提供装置２０間に設置される。ＡＰＩ仕様書生成装置１００は、既存ＡＰＩの仕様を自動で標準仕様の仕様書に変換し生成する。

[0037] ＡＰＩ仕様書生成装置１００は、既存ＡＰＩ開発者２からの入力を受付けるＡＰＩリクエストーレスポンスＩＦ部１０１と、ＡＰＩリクエストーレスポンスキャプチャ部１０２と、ＯＡＳ作成部１１０と、ＯＡＳ雛形設定部１１１と、挿入値ファイル設定部１１２と、パラメータ設定辞書格納部１１３と、を備える。上記ＯＡＳ雛形設定部１１１、挿入値ファイル設定部１１２およびパラメータ設定辞書格納部１１３は、ＡＰＩ記述ルール作成者／システム運用者３からの入力を受付ける。

[0038] また、ＡＰＩ仕様書生成装置１００は、ＡＰＩ記述ルール設定方式入力部１０３と、ＡＰＩ記述ルール作成者／システム運用者３からの入力を受付けるＡＰＩ記述ルール入力部１０４と、ＡＰＩ記述ルール作成者／システム運用者４からの入力を受付けるＡＰＩ記述ルール解析部１０５と、チェックポリシー設定部１０６と、既存ＡＰＩ開発者５からの入力を受付けるＡＰＩ設計エディタ部１０７と、準拠性チェック処理部１０８と、を備える。

なお、既存ＡＰＩ開発者１、既存ＡＰＩ開発者２、および既存ＡＰＩ開発者５は、同一の開発者であってもよい。ＡＰＩ記述ルール作成者／システム運用者３およびＡＰＩ記述ルール作成者／当該システム運用者４は、同一の運用者であってもよい。

[0039] ＡＰＩリクエストーレスポンスＩＦ部１０１は、既存ＡＰＩ開発者２からの入力を変換してＡＰＩ提供装置２０に送信するとともに、ＡＰＩリクエストーレスポンスキャプチャ部１０２に該当ＡＰＩリクエストーレスポンスのデータ取込みを設定する。なお、ＡＰＩ提供装置２０は、ＡＰＩリクエストーレスポンスＩＦ部１０１からの送信に対する応答を返す。

なお、上記ＡＰＩリクエストーレスポンスのデータは、具体的にはＡＰＩリクエストーレスポンス１２６（図１３参照）の各パラメータの値（詳細後記）である。

[0040] ＡＰＩリクエスト－レスポンスキャプチャ部１０２は、ＡＰＩリクエストとそのレスポンスであるＡＰＩリクエスト－レスポンス１２６（図１３参照）をキャプチャする。図２では、ＡＰＩリクエスト－レスポンスキャプチャ部１０２は、Ｌ２スイッチ１５により中継されたＡＰＩリクエスト－レスポンス１２６をキャプチャし、キャプチャしたＡＰＩリクエスト－レスポンス１２６をＯＡＳ作成部１１０に出力する。

[0041] ＯＡＳ作成部１１０は、キャプチャしたＡＰＩリクエスト－レスポンス１２６（図１３参照）データと、挿入値ファイル１２２（図４参照）の参照結果とを雛形ファイル１２１（図３参照）に反映してパラメータを設定し、設定したパラメータをもとにＯＡＳ仕様書１２７（図６参照）を作成する。

[0042] なお、ＡＰＩリクエスト－レスポンス１２６のパラメータに応じて変更する部分は、置換箇所を示す変換タグを記載する。

また、ＯＡＳ作成部１１０は、キャプチャしたＡＰＩリクエスト－レスポンスデータと、挿入値ファイル１２２の参照結果とを雛形ファイル１２１に反映して暫定パラメータを設定するとともに、キャプチャしたＡＰＩリクエスト－レスポンスデータと、パラメータ設定辞書１２３（図５参照）を参照して、暫定パラメータをより精度の高いパラメータに設定し、設定したパラメータをもとにＯＡＳ仕様書１２７を作成する。

[0043] ＯＡＳ作成部１１０は、ＡＰＩリクエスト－レスポンス１２６（図１３参照）のみでは一意に決められず、パラメータ設定辞書１２３の参照が必要とされる領域は、該当するＯＡＳタグ部分に辞書への参照命令を記載する。

なお、挿入値ファイル１２２に記載するパラメータ設定ロジックの書き方によっては、ＨＴＴＰ以外のプロトコルもＯＡＳ仕様書１２７に自動反映することも可能である（後記図８参照）。

[0044] ＯＡＳ雛形設定部１１１は、ＯＡＳ仕様書１２７（図６参照）を作成する際のＯＡＳ雛形（雛形ファイル１２１）（図３参照）をあらかじめ設定する。雛形ファイル１２１は、置換箇所を変換タグとして記載する。

[0045] 挿入値ファイル設定部１１２は、ＯＡＳ仕様書１２７（図６参照）で用い

るパラメータ設定ロジックを記述する挿入値ファイル１２２（図４参照）を設定する。挿入値ファイル１２２は、雛形ファイル１２１に反映するパラメータ設定ロジックを有する。挿入値ファイル設定部１１２は、雛形ファイル１２１（図３参照）記載の変換タグに対応した挿入値ファイル１２２を設定する。

[0046] パラメータ設定辞書格納部１１３は、一意にＯＡＳ仕様書１２７（図６参照）へ反映が難しいパラメータを登録するパラメータ設定辞書１２３（図５参照）を格納する。

[0047] ＡＰＩ記述ルール設定方式入力部１０３は、ＡＰＩ記述ルールの記述様式を入力する。例えば、ＡＰＩ記述ルール作成者／システム運用者４がＡＰＩ記述ルールの記述様式をＡＰＩ記述ルール設定方式入力部１０３にあらかじめ入力する。

[0048] ＡＰＩ記述ルール入力部１０４は、アプリケーションプログラミングインタフェースを設計する際のルールを記述したＡＰＩ記述ルールを入力する。

ＡＰＩ記述ルール入力部１０４は、ＡＰＩ記述ルール作成者／システム運用者３が、アノテーションや正規表現等の一般的なプログラミング言語、または、ＡＰＩ仕様書生成装置１００が指定するＡＰＩ特有の記述様式に従って記載されたＡＰＩ記述ルールを入力するインポート（import）部である。上記ＡＰＩ記述ルールは、ＡＰＩ仕様書生成装置１００が指定する記述様式に従って記載されたSyntax的なＡＰＩ記述ルール、および、Semantic的なＡＰＩ記述ルールである。具体的には、上記Syntax的なエラーは、構文／文法的なエラーである。上記Semantic的なエラーは、ＡＰＩ仕様の設計に関する領域において、使用しているメソッドがＣＲＵＤ（Create, Read, Update, Delete）の考え方に従っているか等意味の違いや矛盾によって生じるエラーである。

[0049] また、ＡＰＩ記述ルール入力部１０４は、ＡＰＩ記述ルール作成者／システム運用者３が、チェックポリシーを記載したソースコード等を入力する。

[0050] ＡＰＩ記述ルール解析部１０５は、ＡＰＩ記述ルールを解析してチェック

ポリシーを作成する。詳細には、ＡＰＩ記述ルール解析部１０５は、ＡＰＩ記述ルール設定方式入力部１０３により事前に規定された記述様式に従い、ＡＰＩ記述ルールを解析してチェックポリシーを作成する。作成したチェックポリシーは、チェックポリシー設定部１０６に送られる。

[0051] チェックポリシー設定部１０６は、ＡＰＩ記述ルール作成者／システム運用者４が、チェックポリシーに間違いのないことを確認する。チェックポリシーに間違いのないことが確認されると、チェックポリシー設定部１０６は、チェックポリシーを準拠性チェック処理部１０８に反映する。

[0052] ＡＰＩ設計エディタ部１０７は、既存ＡＰＩ開発者５が、ＡＰＩ設計書を入力する。

[0053] 準拠性チェック処理部１０８は、チェックポリシーに基づいてＡＰＩ記述ルールに準拠していることをチェックする。準拠性チェック処理部１０８は、ＯＡＳ作成部１１０から送られたＯＡＳ仕様書１２７をもとに、ＡＰＩ記述ルールへの準拠性をチェックする。

具体的には、準拠性チェック処理部１０８は、既存ＡＰＩ開発者５からＡＰＩ設計書の入力を受け付け、入力されたＡＰＩ設計書をチェックポリシーに基づいてチェックし、チェックポリシーに沿っていない非準拠箇所を既存ＡＰＩ開発者５に通知する。

ＡＰＩ記述ルール作成者／システム運用者３は、ＡＰＩ設計エディタ部１０７を用いてＡＰＩ設計書を入力してもよいし、ＡＰＩ仕様書生成装置１００の提供するＡＰＩを用いてＡＰＩ設計書を送付してもよい。ＡＰＩ記述ルール作成者／システム運用者３がＡＰＩ設計エディタ部１０７を用いたときは、ＡＰＩ設計書の非準拠箇所をＡＰＩ設計エディタ部１０７に出力し、既存ＡＰＩ開発者５に通知する。

[0054] なお、図２のＡＰＩリクエスト－レスポンスＩＦ部１０１およびＡＰＩリクエスト－レスポンスキャプチャ部１０２は、図１のキャプチャブロック＃１に対応する。図２のＯＡＳ作成部１１０、ＯＡＳ雛形設定部１１１、挿入値ファイル設定部１１２、およびパラメータ設定辞書格納部１１３は、図１

のＯＡＳ生成処理ブロック＃２に対応する。図２のＡＰＩ記述ルール設定方式入力部１０３、ＡＰＩ記述ルール入力部１０４、ＡＰＩ記述ルール解析部１０５、チェックポリシー設定部１０６、ＡＰＩ設計エディタ部１０７、および準拠性チェック処理部１０８、図１のＯＡＳ出力ブロック＃３に対応する。

[0055] 〈雛形ファイル１２１〉

図３は、ＯＡＳ雛形設定部１１１が設定する雛形ファイル１２１の一例を示す図である。

雛形ファイル１２１は、ＡＰＩリクエストーレスポンス１２６（図１３参照）の各パラメータの値に応じてＯＡＳ仕様書１２７（図６参照）を作成するための雛形である。雛形ファイル１２１は、置換箇所を変換タグとして記載する。

[0056] 雛形ファイル１２１には、ＡＰＩリクエストーレスポンス１２６のパラメータに応じて変更する部分に、置換箇所を示す変換タグが記載される。図３に示す雛形ファイル１２１は、「！！」に示す置換箇所（図３の太文字参照）が変換タグとして記載される。

[0057] 〈挿入値ファイル１２２〉

図４は、挿入値ファイル設定部１１２が設定する挿入値ファイル１２２の一例を示す図である。

挿入値ファイル１２２は、雛形ファイル１２１（図３参照）に反映するパラメータ設定ロジックを有する。なお、一意に特定できないパラメータ領域は、パラメータ設定辞書１２３を参照するなどして決定する。

[0058] 図４の符号ｗに示すように、

・host.yaml

host: [Host:]は、リクエストーレスポンスのHost:ヘッダの内容を、そのまま反映する。

[0059] 図４の符号ｘに示すように、

・basePath.yaml

basePath: [Request URI: - Host:]は、リクエストパラメータの各パラメータを参照してどのようにＯＡＳへ反映するかを設定する。本実施形態では、Request URLヘッダの値からHost:ヘッダの値を差し引いた値をbasePath: として設定するロジックを記載する。

[0060] 図４の符号ｙに示すように、

・type_1.yaml

[Member Key: SELECT ParameterType FROM ParameterＤＢ WHERE [Member Key] = ParameterＤＢ.Word]は、データ型等、ＡＰＩリクエスト－レスポンスの単純解析では、一意に特定不可なパラメータについて、パラメータ設定辞書を参照して設定する。

[0061] 〈パラメータ設定辞書１２３〉

図５は、ＯＡＳ作成部１１０の動作を示す図である。ＯＡＳ作成部１１０の動作については、後記し、パラメータ設定辞書１２３の構成例について述べる。

パラメータ設定辞書１２３は、一意にＯＡＳ仕様書１２７（図６参照）へ反映が難しいパラメータを登録する。

図５に示すように、パラメータ設定辞書１２３は、Word(項目)に対応するType(型)、format(フォーマット)、maximum(最大値)、およびminimum(最小値)を有する。Wordは、例えば、Birthday、Telephone、Creditである。Typeは、例えば、integer(整数)である。formatは、int32(整数32bit)、int64(整数64bit)である。

[0062] 図５に示すパラメータ設定辞書１２３で、例えば、Word「Birthday」が検索されると、Type 「integer」、format「int32」、maximum「99999999」、およびminimum「00」が参照される。Word「Birthday」の場合、maximumは、yyyymmdd(西暦4桁,月2桁,日2桁)が設定され、minimumは、dd(日2桁)が設定されている。このため、minimumに「00」(2桁)より大きいパラメータは設定できない。

同様に、Word「Telephone」が検索されると、Type 「integer」、format「

int32」、maximum「－(設定なし)」、およびminimum「0000000000」(8桁)が参照される。

[0063] ＯＡＳ作成部１１０は、パラメータ設定辞書１２３を参照することで、リクエスト－レスポンスの解析のみでは機械的に設定できないパラメータついても、高い精度で適切なパラメータを設定することが可能となる（後記）。

[0064] 以下、上述のように構成されたＡＰＩ仕様書生成装置１００のＡＰＩ仕様書生成方法について説明する。

［全体動作］

ＳＴＥＰ１

図２の符号ｈに示すように、既存ＡＰＬ１１からＡＰＩ提供装置２０に対してＡＰＩリクエストを送信する。

[0065] ＳＴＥＰ２

ＡＰＬリクエストがない場合、図２の符号ｉに示すように、既存ＡＰＩ開発者２は、既存ＡＰＩ１１にＡＰＩリクエストを送信するために必要な情報（ＵＲＬ，リクエストパラメータ等）を入力する。既存ＡＰＩ開発者２は、ＡＰＩリクエストに対するレスポンス結果も確認する。

[0066] ＳＴＥＰ３

図２の符号ｊに示すように、ＡＰＩリクエスト－レスポンスキャプチャ部１０２は、Ｌ２スイッチ１５により中継されたＡＰＩリクエスト－レスポンス１２６（図１３参照）をキャプチャする。

[0067] ＳＴＥＰ４

図２の破線矢印および符号ｋに示すように、ＡＰＩ記述ルール作成者／システム運用者３は、ＯＡＳ雛形設定部１１１の雛形ファイル１２１、挿入値ファイル設定部１１２の挿入値ファイル１２２、パラメータ設定辞書格納部１１３のパラメータ設定辞書１２３について、手動により雛形ファイル１２１、挿入値ファイル１２２、パラメータ設定辞書１２３を充実化またはカスタマイズさせる。

[0068] ＳＴＥＰ５

図２の符号ｌに示すように、ＯＡＳ作成部１１０は、ＯＡＳ雛形設定部１１１に設定された雛形ファイル１２１（図３参照）に、ＡＰＩリクエスト－レスポンスキャプチャ部１０２でキャプチャされたＡＰＩリクエスト－レスポンス１２６のキャプチャデータと挿入値ファイル設定部１１２の参照結果とを反映させる。

[0069] ＳＴＥＰ６

図２の符号ｍに示すように、ＯＡＳ作成部１１０は、一意にＯＡＳへ反映が難しいパラメータは、パラメータ設定辞書格納部１１３が格納するパラメータ設定辞書１２３（図５参照）を参照して、暫定的にパラメータを設定する。

[0070] ＳＴＥＰ７

図２の符号ｎに示すように、ＡＰＩ記述ルール作成者／システム運用者３は、ＡＰＩ記述ルール入力部１０４に、アノテーションや正規表現等の一般的なプログラミング言語、または、ＡＰＩ仕様書生成装置１００が指定するＡＰＩ特有の記述様式に従って記載されたＡＰＩ記述ルールを入力する。ＡＰＩ記述ルール作成者／システム運用者３は、ＡＰＩ記述ルール設定方式入力部１０３（後記）で設定された記述様式に従ってＡＰＩ記述ルールを記載する。

[0071] 上記ＡＰＩ記述ルールは、ＡＰＩ仕様書生成装置１００が指定する記述様式に従って記載されたSyntax的なＡＰＩ記述ルール、および、Semantic的なＡＰＩ記述ルールである。具体的には、ＡＰＩ記述ルールは、ＡＰＩのリソース名、パラメータ名に使用してはいけない文字や文字列を規定したルール、およびＡＰＩで使用するメソッドの使用用途を規定したルールなどである。メソッドの使用用途は、ＲＥＳＴ ＡＰＩ１０のメソッドが限定的な特徴を活かしてルールを規定できる。

また、ＡＰＩ記述ルール作成者／システム運用者３は、チェックポリシーを記載したソースコード等を入力する。

[0072] ＳＴＥＰ８

ＡＰＩ記述ルール設定方式入力部１０３は、ＡＰＩ記述ルールの記述様式を入力する。このＡＰＩ記述ルールの記述様式は、例えばＡＰＩ記述ルール作成者／システム運用者４があらかじめ入力する。

[0073] ＳＴＥＰ９

図２の符号ｏに示すように、ＡＰＩ記述ルール解析部１０５は、ＡＰＩ記述ルール設定方式入力部１０３により事前に規定された記述様式に従い、ＡＰＩ記述ルールを解析してチェックポリシーを決定する。決定されたチェックポリシーは、チェックポリシー設定部１０６に送られる。

[0074] ＳＴＥＰ１０

図２の符号ｐに示すように、ＡＰＩ記述ルール作成者／システム運用者４は、チェックポリシー設定部１０６で、チェックポリシーに間違いのないことを確認する。チェックポリシーに間違いのないことが確認されると、チェックポリシー設定部１０６は、チェックポリシーを準拠性チェック処理部１０８に反映する（図２の符号ｑ参照）。

[0075] ＳＴＥＰ１１

一方、図２の符号ｒに示すように、ＯＡＳ作成部１１０は、作成したＯＡＳをＡＰＩ設計エディタ部１０７へ入力する。ＡＰＩ設計エディタ部１０７は、ＯＡＳとして作成された内容を表示する。

[0076] ＳＴＥＰ１２

図２の符号ｓに示すように、ＡＰＩ設計エディタ部１０７は、ＯＡＳ作成部１１０により作成されたＯＡＳを表示させ、既存ＡＰＩ開発者５によるＡＰＩ設計書の設計および記載を支援する。

[0077] ＳＴＥＰ１２

図２の符号ｔに示すように、準拠性チェック処理部１０８は、既存ＡＰＩ開発者５からＡＰＩ設計書（ＯＡＳ）の入力を受け付け、入力されたＡＰＩ設計書（ＯＡＳ）をチェックポリシーに基づいてチェックし、チェックポリシーに沿っていない非準拠箇所を既存ＡＰＩ開発者５に通知する（図２の符号ｕ参照）。既存ＡＰＩ開発者５は、ＡＰＩ設計エディタ部１０７を用いて

ＡＰＩ設計書（ＯＡＳ）を入力する。

[0078] 特に、図２の符号ｖに示すように、ＡＰＩ設計エディタ部１０７を経由して、準拠性チェック処理部１０８にＯＡＳを送ることで、準拠性チェック処理部１０８は、特定のＡＰＩ記述ルールへの準拠性チェックも確認可能である。なお、特定のＡＰＩ記述ルールは、ＡＰＩ記述ルール設定方式入力部１０３で入力されているＡＰＩ記述ルールの記述様式では準拠性チェックできないものを含む。

準拠性チェック処理部１０８は、既存ＡＰＩ開発者５に特定のＡＰＩ記述ルールに準拠した設計内容を提案する（図２の符号ｕ参照）。

[0079] ［ＯＡＳ作成部１１０動作］

図５を参照してＯＡＳ作成部１１０の動作を説明する。

ＯＡＳ作成部１１０（図２参照）は、ＡＰＩリクエスト－レスポンスキャプチャ部１０２（図２参照）でキャプチャされたＡＰＩリクエスト－レスポンス１２６のデータをパラメータ設定ロジックに従い、挿入値ファイル１２２へ反映させる。

図５に示す挿入値ファイル１２２において、

・host.yaml「host:」「basePath:」は、パラメータ設定ロジックに設定されている。このため、図５の符号ｚに示すように、・host.yamlは、ＡＰＩリクエスト１２４（図１３参照）の host: petstore-api.herokuapp.comそのままとなる。同様に、・basePath.yamlは、basePath: /Petとなる。

[0080] ＯＡＳ作成部１１０は、キャプチャしたＡＰＩリクエスト－レスポンス１２６のデータと、挿入値ファイル１２２の参照結果とを雛形ファイル１２１に反映して暫定パラメータを設定する。

[0081] 一方、「・name_1.yaml」「・type_1.yaml」「・format_1.yaml」は、パラメータ設定ロジックに設定されていない。このため、図５の符号ａ１に示すように、・host.yamlは、ＡＰＩリクエスト－レスポンスの解析のみで機械的に設定できないパラメータ（一意に特定不可であるパラメータ）である。

この場合は、図５の符号ｂ１に示すように、パラメータ設定辞書１２３を

参照して、「・name_1.yaml」「・type_1.yaml」「・format_1.yaml」に対して各パラメータを設定し、挿入値ファイル１２２に設計を反映する。

[0082] 上述したように、図５に示すパラメータ設定辞書１２３は、Wordに対応するType、format、maximum、およびminimumを有する。図５の符号ｂ１に示すように、Word「Birthday」が検索されると、Type 「integer」、format「int32」、maximum「99999999」、およびminimum「00」が参照され、挿入値ファイル１２２に設定される。すなわち、

・name_1.yaml

birthday:

・type_1.yaml

type: integer

・format_1.yaml

format: int32

が挿入値ファイル１２２に設定される。

[0083] ＯＡＳ作成部１１０は、キャプチャしたＡＰＩリクエストーレスポンス１２６のデータと、パラメータ設定辞書１２３を参照して、暫定パラメータをより精度の高いパラメータに設定する。すなわち、一意にＯＡＳへ反映が難しいパラメータは、パラメータ設定辞書１２３を参照して、より精度の高いパラメータを設定する。

[0084] 図６は、図５のＯＡＳ作成部１１０の動作により作成されたＯＡＳ仕様書１２７を示す図である。

図６に示すＯＡＳ仕様書１２７は、雛形ファイル１２１（図３参照）に挿入値ファイル１２２（図５参照）を挿入済のファイルを示す。

図６中の太文字は、ＡＰＩリクエストーレスポンス１２６のキャプチャデータとパラメータ設定辞書１２３を参照し、雛形ファイル１２１（図３参照）へ設計を反映した箇所である。

例えば、図６に示すＯＡＳ仕様書１２７は、雛形ファイル１２１（図３参照）をもとに、

swagger: '2.0'

info:

version: 1.0.0

title: hogehoge

description: |

を記述し、この記述に続いて、図５に示す挿入値ファイル１２２から下記パラメータを挿入する。

host: petstore-api.herokuapp.com

basePath: /pet

schemes:

- http

consumes:

- application/json

produces:

- application/json

paths:

/:

[0085] 続いて、図５に示す挿入値ファイル１２２から下記パラメータを挿入する。

post:

parameters:

[0086] 雛形ファイル１２１（図３参照）をもとに、

－ name: A

を記述し、この記述に続いて、図５に示す挿入値ファイル１２２から下記パラメータを挿入する。

in: body

[0087] 雛形ファイル１２１（図３参照）をもとに、

description: hogehogeschema:

を記述し、この記述に続いて、図５に示す挿入値ファイル１２２から下記パラメータを挿入する。

```
$ref: '#／definitions／A'
```

[0088] 雛形ファイル１２１（図３参照）をもとに、

```
required: true
```

を記述し、この記述に続いて、図５に示す挿入値ファイル１２２から下記パラメータを挿入する。

```
responses:
  200:
```

[0089] 雛形ファイル１２１（図３参照）をもとに、

```
description: hogehoge
```

を記述する。

[0090] 以上で、「post:」の記述とその挿入値ファイル１２２からのパラメータ挿入を終え、「name: A」についての定義を記述する。すなわち、

雛形ファイル１２１（図３参照）をもとに、

```
definitions:
  A:
```

を記述し、この記述に続いて、図５に示す挿入値ファイル１２２から下記パラメータを挿入する。

```
type: object
```

[0091] 続いて、雛形ファイル１２１（図３参照）をもとに、

```
properties:
```

を記述し、この記述に続いて、図５に示す挿入値ファイル１２２から下記パラメータを挿入する。

```
name:
  type: string
birthday:
  type: integer
```

format: int32

上記パラメータは、一意にＯＡＳへ反映が難しいパラメータであり、パラメータ設定辞書１２３を参照して設定される。

[0092] 以上、ＨＴＴＰ形式のＲＥＳＴ ＡＰＩを用いたＯＡＳ作成部１１０の動作について説明した。

[0093] ［変形例］

次に、変形例として、ＳＩＰ（Session Initiation Protocol）をＯＡＳ変換ロジックに変える場合について説明する。

[0094] 〈雛形ファイル１２１Ａ〉

図７は、ＳＩＰをＨＴＴＰ形式のＲＥＳＴ ＡＰＩに変換する場合、ＯＡＳ雛形設定部１１１が設定する雛形ファイル１２１Ａの一例を示す図である。図３の雛形ファイル１２１と同一部分には、同一パラメータを付している。

雛形ファイル１２１Ａは、ＡＰＩリクエスト－レスポンス１２６（図１３参照）の各パラメータの値に応じてＯＡＳ仕様書を作成するための雛形である。

雛形ファイル１２１Ａには、ＡＰＩリクエスト－レスポンス１２６のパラメータに応じて変更する部分に、置換箇所を示す変換タグが記載される。

[0095] ＯＡＳ作成部１１０（図５参照）は、雛形ファイル１２１Ａ記載の変換タグに対応した挿入値ファイル１２２Ａ（図８参照）を設定する。

[0096] 図７に示す雛形ファイル１２１Ａは、

前記図３に示す雛形ファイル１２１の下記パラメータ

```
properties:
  ！！Conversion name_1:
    ！！Conversion type_1:
    ！！Conversion format_1
  ！！Conversion name_2:
    ！！Conversion type_2:
    ！！Conversion format_2:
```

が、下記パラメータ

properties:

from

！！Conversion type_1:

！！Conversion format_1

to

！！Conversion type_2:

！！Conversion format_2:

に置き換えられている。

[0097] 〈挿入値ファイル１２２Ａ〉

図８は、挿入値ファイル設定部１１２が設定する挿入値ファイル１２２Ａの一例を示す図である。

挿入値ファイル１２２Ａは、雛形ファイル１２１Ａ（図７参照）に反映するパラメータロジックを有するファイルである。なお、一意に特定できないパラメータ領域は、パラメータ設定辞書を参照するなどして決定する。

図８に示す挿入値ファイル１２２Ａは、

・host.yaml

host: [REGISTER sip:]

・method.yaml

－ [REGESTER: post]

－ [INVITE: get]

・type_1.yaml

[To:]

を有する。

[0098] 図８の符号ｃ１に示すように、

・host.yaml

host: [REGISTER sip:]は、網内プロキシに登録要求をするREGESTERメソッドによるリクエストのSIP URIをＨＴＴＰのhost部とする場合の例である。

[0099] 図８の符号ｄ１に示すように、

・method.yaml

－ [REGESTER: post]

－ [INVITE: getは、SIP REGESTERをＨＴＴＰのPOSTメソッド、SIP INVITEをGETメソッドとする場合の例である。

[0100] 図８の符号ｅ１に示すように、

・type_1.yaml

[To:]は、SIP INVITEのTo:ヘッダの値を反映する場合の例である。

[0101] 以上、ＳＩＰをＨＴＴＰ形式のＲＥＳＴ ＡＰＩに変換する場合、ＳＩＰリクエスト－レスポンスの値を基に、ＨＴＴＰプロトコルへ変換するロジックを反映する。これにより、ＯＡＳ形式の仕様書を自動生成することが可能になる。

[0102] [構文的記述ルールの規定例]

次に、構文的記述ルールの規定例について説明する。

図９は、ＡＰＩ仕様書生成装置１００が規定する注釈様式に従って、構文的記述ルールを記載した例を示す図である。

[0103] 図９の例では、ＡＰＩのリソース名、パラメータ名に、アプリが動作する上でバグの原因となるおそれのある文字「％」，「：」，「￥」，「￥￥」，「＼（バックスラッシュ）」の使用を禁止するルールが記載されている。

具体的には、図９の符号ｆ１に示すように、“@NGcheck¥Resource”は、ＡＰＩ仕様書生成装置１００が規定した記述様式であり、ＡＰＩ設計書内の各ＡＰＩのリソース名内部で特定の記述等がされた場合にエラーを出力するルール（ポリシー）である。また、図９の符号ｇ１に示すように、“@NGcheck¥Parameters”は、各ＡＰＩのパラメータ名に特定の文字、文字列が存在する場合にエラー情報を出力するルール（ポリシー）である。また、図９の符号ｈ１に示すように、使用してはいけない文字、文字列が正規表現等で記述されている。

[0104] このように、ＡＰＩのリソース名、パラメータ名に、アプリ動作する上で

バグの原因となるおそれのある「%」，「:」，「¥」，「＼」の使用を禁止する場合は、ＡＰＩ仕様書生成装置１００が指定する記述様式を用いて正規表現等を活用して記述したファイルをＡＰＩ記述ルール入力部１０４（図２参照）にインポートする（図９の符号ｉ１参照）。

[0105] 準拠性チェック処理部１０８（図２参照）は、記述された内容に従って、準拠性チェックポリシーを規定する。

図９の符号ｊ１に示すように、ＡＰＩ設計エディタ部１０７（図２参照）は、記載されたＡＰＩ仕様を、事前に規定されたチェックポリシーに従って準拠性確認をし、その結果を既存ＡＰＩ開発者５に通知する。通知形式は、ＡＰＩ設計エディタ部１０７がリアルタイムに通知を行う。

[0106] ［メソッド使用ルールの規定例］

次に、メソッド使用ルールの規定例について説明する。

図１０は、ＡＰＩ記述ルールを表にして示す図である。図１０に示す表は、ＡＰＩ記述ルールとして各メソッドの使用用途を規定する。

図１０の表に示すように、ＣＲＵＤ（Create, Read, Update, Delete）毎に、メソッド名（ＰＯＳＴ，ＧＥＴ，ＰＵＴ，ＤＥＬＥＴＥ）と、対応する用途を規定する。

ＡＰＩ記述ルールとして各メソッドの使用用途を図１０の表に規定したと仮定する。例えば、リソースを新規に作成／更新／削除するようなＡＰＩにもかかわらず、「ＧＥＴ」を用いているかを確認する。図１０の表で規定したルールに反して用途に合っていないメソッドが使われている場合にエラーを出力する。

[0107] 図１１は、ＡＰＩ仕様書生成装置１００が規定する記述様式に従って、メソッド使用ルールを記載した例を示す図である。

[0108] 図１１の符号ｋ１に示すように、“@NGcheck¥Api”は、一つのエンドポイント毎のＡＰＩに関する設計（概要やメソッド）において、特定の条件に当てはまる記述が存在する場合にエラーを出力するルール（ポリシー）である。図１１の符号Ｉ１に示すように、ＡＰＩ仕様書生成装置１００が規定した

記述様式でエラーを出力するポリシーを記載する。例えば、ＡＰＩの説明文（description部）に「更新」、「create」、「追加」、「add」、「削除」、「delete」という記載があり、かつ、ＧＥＴメソッドを使用している場合、エラーを出力する。より具体的には、ＡＰＩ設計書において、ある行にＧＥＴメソッドを使用することが記載されており、その後の行のＡＰＩの説明文に“Create”の記述がある場合を想定する。このＡＰＩ設計書は、リソースを新規に作成するＡＰＩにもかかわらずＧＥＴを用いている。したがって、このＡＰＩ設計書をＡＰＩ仕様書生成装置１００に入力した場合、用途に合っていないメソッドが使われている旨のエラー情報が出力される。

ＡＰＩ仕様書生成装置１００が規定する記述様式に沿った設計、または、一般的なプログラミング言語で、エラー検出のロジックを組むことで、ルール違反したメソッドの使用をチェックすることが可能になる。

[0109] 以上説明したように、本実施形態のＡＰＩ仕様書生成装置１００（図１参照）は、ＯＡＳ仕様書１２７を作成する際の雛形ファイル１２１をあらかじめ設定するＯＡＳ雛形設定部１１１と、ＯＡＳ仕様書１２７で用いるパラメータ設定ロジックを記述する挿入値ファイル１２２を設定する挿入値ファイル設定部１１２と、ＡＰＩリクエストとそのレスポンスをキャプチャするＡＰＩリクエスト－レスポンスキャプチャ部１０２と、キャプチャしたＡＰＩリクエスト－レスポンスデータと、挿入値ファイル１２２の参照結果とを雛形ファイル１２１に反映してパラメータを設定し、設定したパラメータをもとにＯＡＳ仕様書１２７を作成するＯＡＳ作成部１１０と、を備える。

[0110] この構成により、ＡＰＩリクエスト－レスポンス１２６をもとに低コストでデータ型や値の範囲まで反映するＯＡＳ仕様書１２７を作成できる。また、各種プロトコルの値から、ＯＡＳにパラメータをどのように設定するかを記述するパラメータ設定ロジックも、システム運用者にてカスタマイズ可能である。さらに、独自パラメータやＨＴＴＰ以外のプロトコルもＯＡＳ仕様書の対象にすることができる。

[0111] 本実施形態では、ＡＰＩ仕様書生成装置１００は、一意にＯＡＳ仕様書１

２７へ反映が難しいパラメータを登録するパラメータ設定辞書１２３を格納するパラメータ設定辞書格納部１１３を備え、ＯＡＳ作成部１１０は、キャプチャしたＡＰＩリクエスト－レスポンス１２６のデータと、挿入値ファイル１２２の参照結果とを雛形ファイル１２１に反映して暫定パラメータを設定するとともに、キャプチャしたＡＰＩリクエスト－レスポンス１２６のデータと、パラメータ設定辞書を参照して、暫定パラメータをより精度の高いパラメータに設定し、設定したパラメータをもとにＯＡＳ仕様書１２７を作成する。

[0112] このようにすることで、ＡＰＩリクエスト－レスポンス１２６とＡＰＩパラメータ設定辞書１２３を参照して、一意に特定不可であるパラメータも高精度で設計可能となる。このため、ＯＡＳ仕様書１２７の作成および修正稼働を削減することができる。

[0113] また、上記実施形態において説明した各処理のうち、自動的に行われるものとして説明した処理の全部または一部を手動的に行うこともでき、あるいは、手動的に行われるものとして説明した処理の全部または一部を公知の方法で自動的に行うこともできる。この他、上述文書中や図面中に示した処理手順、制御手順、具体的名称、各種のデータやパラメータを含む情報については、特記する場合を除いて任意に変更することができる。

また、図示した各装置の各構成要素は機能概念的なものであり、必ずしも物理的に図示の如く構成されていることを要しない。すなわち、各装置の分散・統合の具体的形態は図示のものに限られず、その全部または一部を、各種の負荷や使用状況などに応じて、任意の単位で機能的または物理的に分散・統合して構成することができる。

[0114] また、上記の各構成、機能、処理部、処理手段等は、それらの一部または全部を、例えば集積回路で設計する等によりハードウェアで実現してもよい。また、上記の各構成、機能等は、プロセッサがそれぞれの機能を実現するプログラムを解釈し、実行するためのソフトウェアで実現してもよい。各機能を実現するプログラム、テーブル、ファイル等の情報は、メモリや、ハー

ドディスク、ＳＳＤ（Solid State Drive）等の記録装置、または、ＩＣ（Integrated Circuit）カード、ＳＤ（Secure Digital）カード、光ディスク等の記録媒体に保持することができる。また、本明細書において、時系列的な処理を記述する処理ステップは、記載された順序に沿って時系列的に行われる処理はもちろん、必ずしも時系列的に処理されなくとも、並列的あるいは個別に実行される処理（例えば、並列処理あるいはオブジェクトによる処理）をも含むものである。

符号の説明

[0115] １０　ＲＥＳＴ　ＡＰＩ

１１　既存ＡＰＬ

１５　Ｌ２スイッチ

２０　ＡＰＩ提供装置（既存機能）

１００　ＡＰＩ仕様書生成装置

１０１　ＡＰＩリクエスト－レスポンスＩＦ部

１０２　ＡＰＩリクエスト－レスポンスキャプチャ部（ＡＰＩリクエスト－レスポンスキャプチャ手段）

１０３　ＡＰＩ記述ルール設定方式入力部

１０４　ＡＰＩ記述ルール入力部

１０５　ＡＰＩ記述ルール解析部

１０６　チェックポリシー設定部

１０７　ＡＰＩ設計エディタ部

１０８　準拠性チェック処理部

１１０　ＯＡＳ作成部（ＯＡＳ作成手段）

１１１　ＯＡＳ雛形設定部（ＯＡＳ雛形設定手段）

１１２　挿入値ファイル設定部（挿入値ファイル設定手段）

１１３　パラメータ設定辞書格納部

１２１，１２１Ａ　雛形ファイル

１２２，１２２Ａ　挿入値ファイル

１２３　パラメータ設定辞書

１２４　ＡＰＩリクエスト

１２５　ＡＰＩレスポンス

１２６　ＡＰＩリクエストーレスポンス

１２７　ＯＡＳ仕様書

請求の範囲

[請求項1] 既存ＡＰＩ（Application Program Interface）の仕様を変換して標準仕様のＯＡＳ（Open API spec）仕様書を生成するＡＰＩ仕様書生成装置であって、

前記ＯＡＳ仕様書を作成する際の雛形ファイルをあらかじめ設定するＯＡＳ雛形設定部と、

前記ＯＡＳ仕様書で用いるパラメータ設定ロジックを記述する挿入値ファイルを設定する挿入値ファイル設定部と、

ＡＰＩリクエストとそのレスポンスをキャプチャするＡＰＩリクエスト－レスポンスキャプチャ部と、

キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記挿入値ファイルの参照結果とを前記雛形ファイルに反映してパラメータを設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成するＯＡＳ作成部と、を備える

ことを特徴とするＡＰＩ仕様書生成装置。

[請求項2] 一意にＯＡＳ仕様書へパラメータを登録するパラメータ設定辞書を格納するパラメータ設定辞書格納部をさらに備え、

前記ＯＡＳ作成部は、

キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記挿入値ファイルの参照結果とを前記雛形ファイルに反映して暫定パラメータを設定するとともに、

キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記パラメータ設定辞書を参照して、前記暫定パラメータを、当該暫定パラメータよりも精度の高いパラメータに設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成する

ことを特徴とする請求項１に記載のＡＰＩ仕様書生成装置。

[請求項3] アプリケーションプログラミングインタフェースを設計する際のルールを記述したＡＰＩ記述ルールを入力するＡＰＩ記述ルール入力部

と、

前記ＡＰＩ記述ルールを解析してチェックポリシーを作成するＡＰＩ記述ルール解析部と、

前記チェックポリシーに基づいて前記ＡＰＩ記述ルールに準拠していることをチェックする準拠性チェック処理部と、をさらに備え、

前記ＯＡＳ作成部は、

作成した前記ＯＡＳ仕様書を前記準拠性チェック処理部に送り、

前記準拠性チェック処理部は、

前記ＯＡＳ作成部から送られた前記ＯＡＳ仕様書をもとに、前記ＡＰＩ記述ルールへの準拠性をチェックする

ことを特徴とする請求項１に記載のＡＰＩ仕様書生成装置。

[請求項4] 既存ＡＰＩの仕様を変換して標準仕様のＯＡＳ仕様書を生成するＡＰＩ仕様書生成装置が、

前記ＯＡＳ仕様書を作成する際の雛形ファイルをあらかじめ設定するステップと、

前記ＯＡＳ仕様書で用いるパラメータ設定ロジックを記述する挿入値ファイルを設定するステップと、

ＡＰＩリクエストとそのレスポンスをキャプチャするステップと、

キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記挿入値ファイルの参照結果とを前記雛形ファイルに反映してパラメータを設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成するステップと、を実行する

ことを特徴とするＡＰＩ仕様書生成方法。

[請求項5] 既存ＡＰＩの仕様を変換して標準仕様のＯＡＳ仕様書を生成するＡＰＩ仕様書生成装置としてのコンピュータを、

前記ＯＡＳ仕様書を作成する際の雛形ファイルをあらかじめ設定するＯＡＳ雛形設定手段、

前記ＯＡＳ仕様書で用いるパラメータ設定ロジックを記述する挿入

値ファイルを設定する挿入値ファイル設定手段、

ＡＰＩリクエストとそのレスポンスをキャプチャするＡＰＩリクエスト－レスポンスキャプチャ手段、

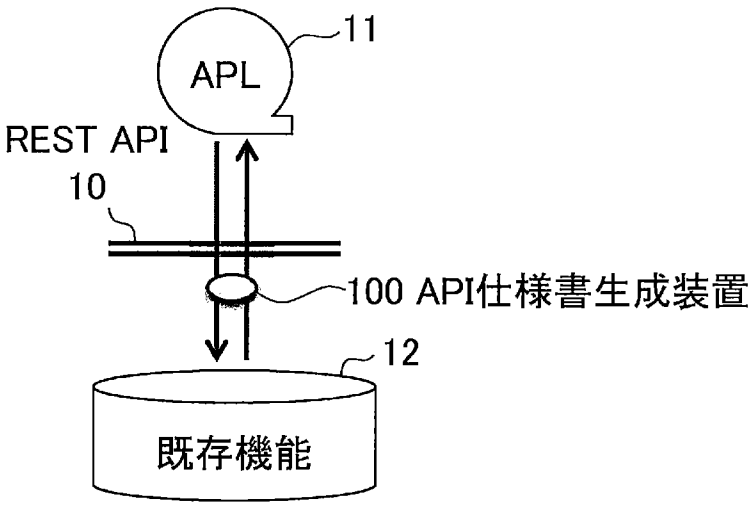
キャプチャしたＡＰＩリクエスト－レスポンスデータと、前記挿入値ファイルの参照結果とを前記雛形ファイルに反映してパラメータを設定し、設定した前記パラメータをもとに前記ＯＡＳ仕様書を作成するＯＡＳ作成手段、

として機能させるためのプログラム。

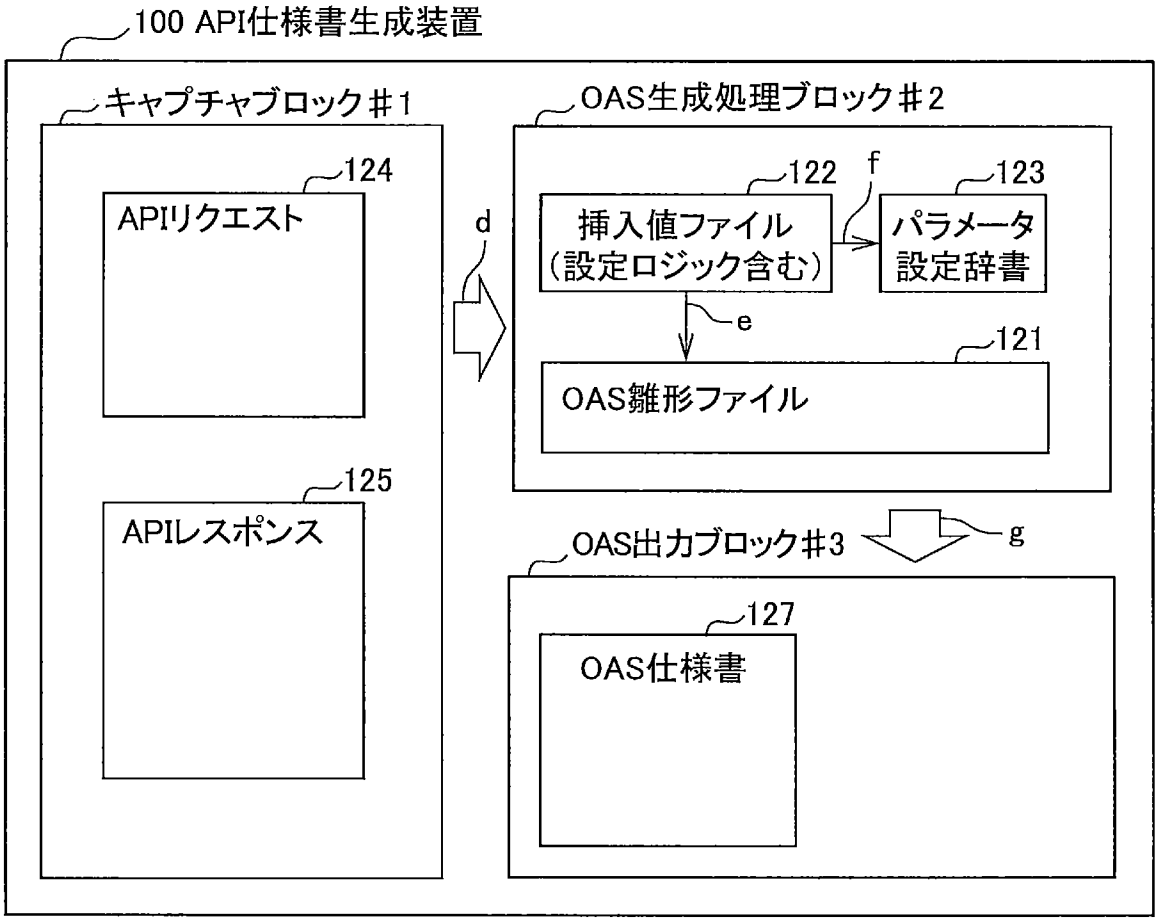
WO 2020/031845 PCT/JP2019/030228

[図1]

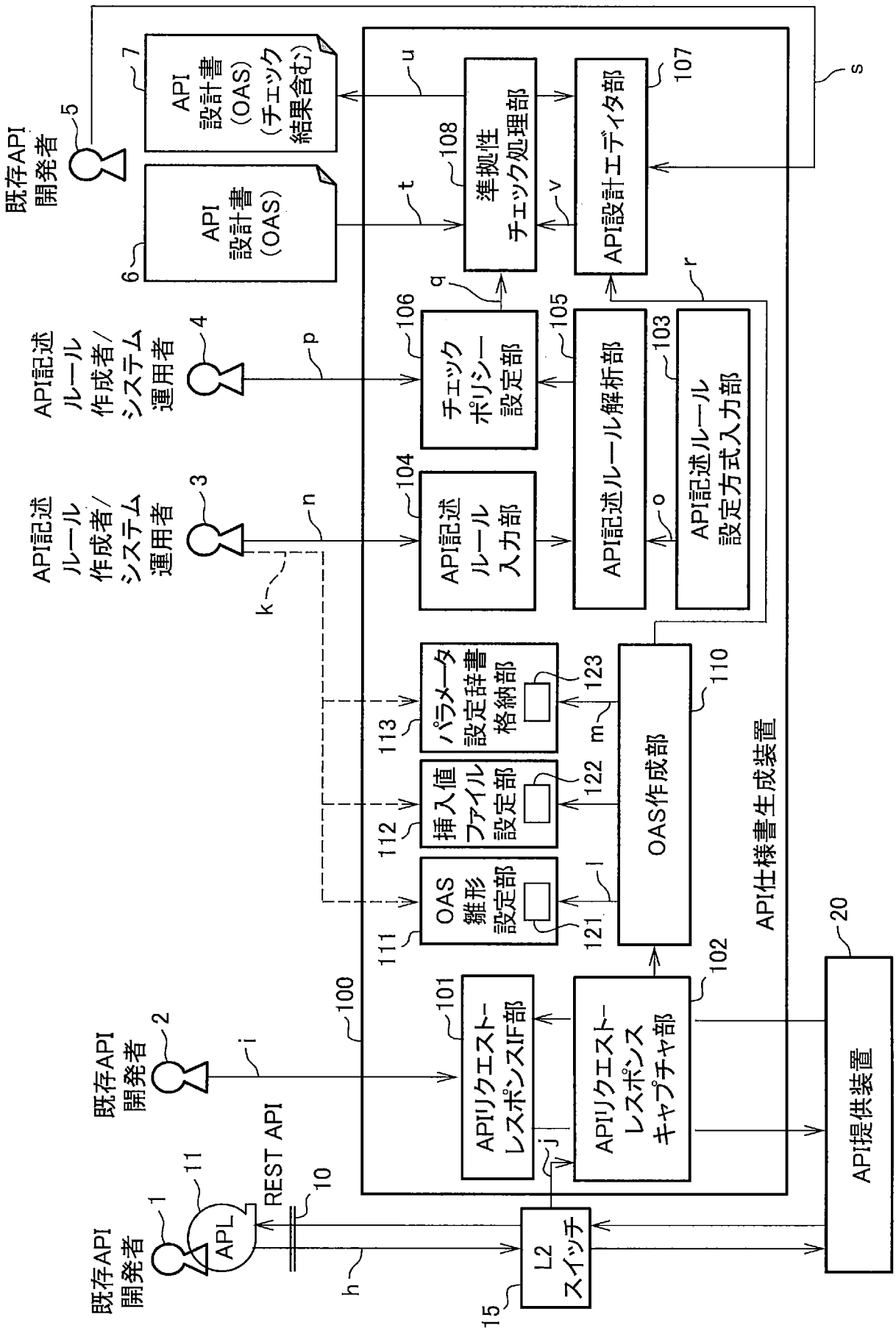
(a)



(b)



[図2]

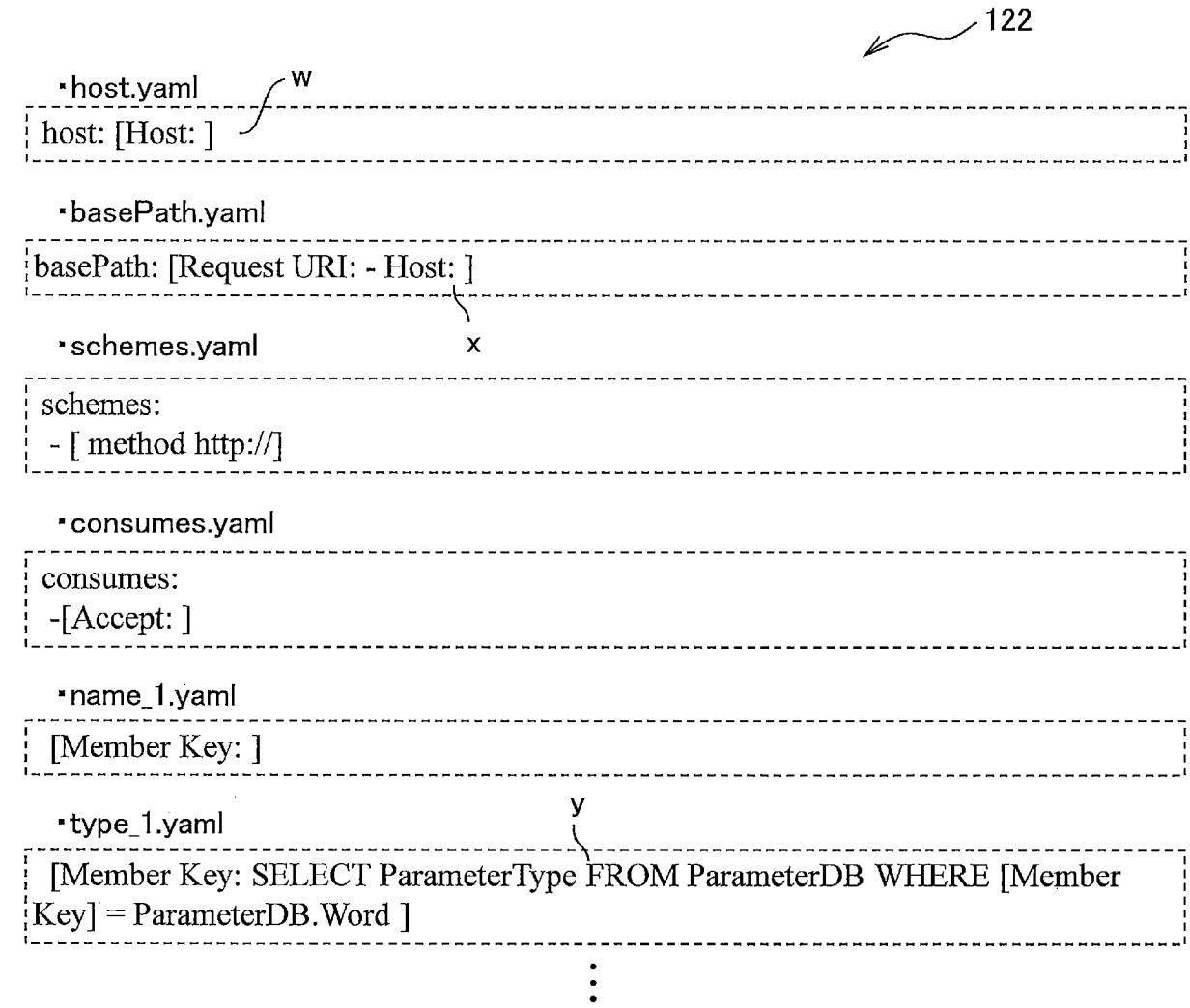

既存API
開発者
5
API
設計書
(OAS)
(チェック
結果含む)
7
API
設計書
(OAS)
6
準拠性
チェック処理部
108
API設計エディタ部
107
API記述
ルール
作成者/
システム
運用者
4
チェック
ポリシー
設定部
106
API記述
ルール
作成者/
システム
運用者
3
API記述
ルール
入力部
104
API記述ルール解析部
105
API記述ルール
設定方式入力部
103
パラメータ
設定辞書
格納部
123
挿入値
ファイル
設定部
122
OAS
雛形
設定部
121
OAS作成部
110
既存API
開発者
2
既存API
開発者
1
APL
11
REST API
10
APIリクエスト-
レスポンスIF部
101
APIリクエスト-
レスポンス
キャプチャ部
102
L2
スイッチ
15
API提供装置
20
API仕様書生成装置
100
111
112
113
h
i
j
k
l
m
n
o
p
q
r
s
t
u
v

[図3]

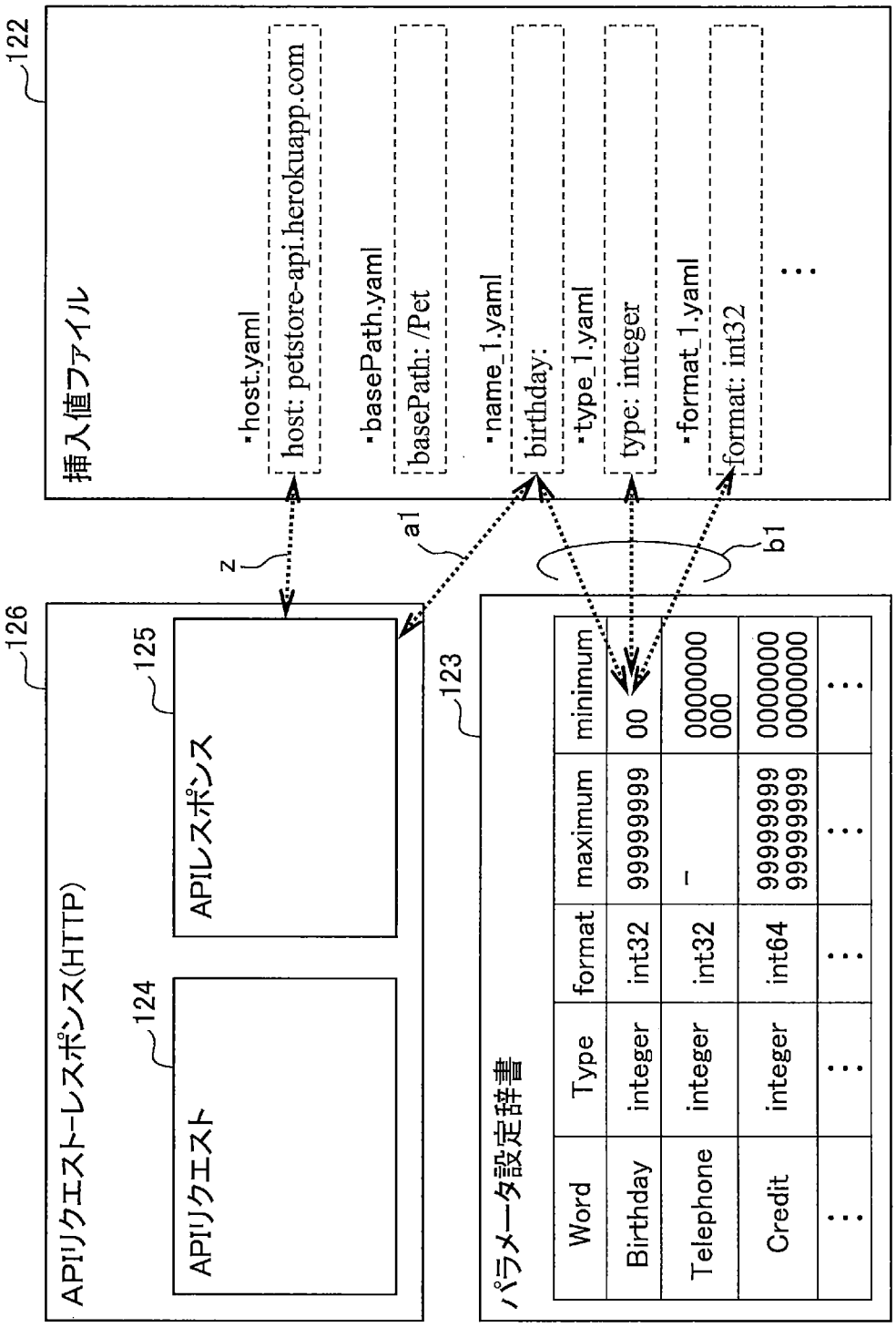
121

```
swagger: '2.0'
info:
  version: 1.0.0
  title: hogehoge
  description: |hogehoge

  ! ! Conversion host:
  ! ! Conversion basePath:
  ! ! Conversion schemes:
  ! ! Conversion consumes:

  ! ! Conversion produces:
  ! ! Conversion Paths:

    ! ! Conversion method:
    parameters:
      - name: X
        ! ! Conversion in:
        description: hogehoge
        ! ! Conversion schema:
        required: true
      ! ! Conversion responses:
        ! ! Conversion response code:
        description: hogehoge

definitions:
  A:
    ! ! Conversion JavaScript Object:
    properties:
      ! ! Conversion name_1:
        ! ! Conversion type_1:
        ! ! Conversion format_1
      ! ! Conversion name_2:
        ! ! Conversion type_2:
        ! ! Conversion format_2:
```

[図4]


122
・host.yaml
w
host: [Host:]
・basePath.yaml
basePath: [Request URI: - Host:]
x
・schemes.yaml
schemes:
- [method http://]
・consumes.yaml
consumes:
-[Accept:]
・name_1.yaml
[Member Key:]
・type_1.yaml
y
[Member Key: SELECT ParameterType FROM ParameterDB WHERE [Member Key] = ParameterDB.Word]

[図5]


122
挿入値ファイル
・host.yaml
host: petstore-api.herokuapp.com
・basePath.yaml
basePath: /Pet
・name_1.yaml
birthday:
・type_1.yaml
type: integer
・format_1.yaml
format: int32
z
a1
b1
126
APIリクエストレスポンス(HTTP)
125
APIレスポンス
124
APIリクエスト
123
パラメータ設定辞書
Word
Type
format
maximum
minimum
Birthday
integer
int32
99999999
00
Telephone
integer
int32
-
0000000000
Credit
integer
int64
9999999999999999
00000000000000

[図6]

127

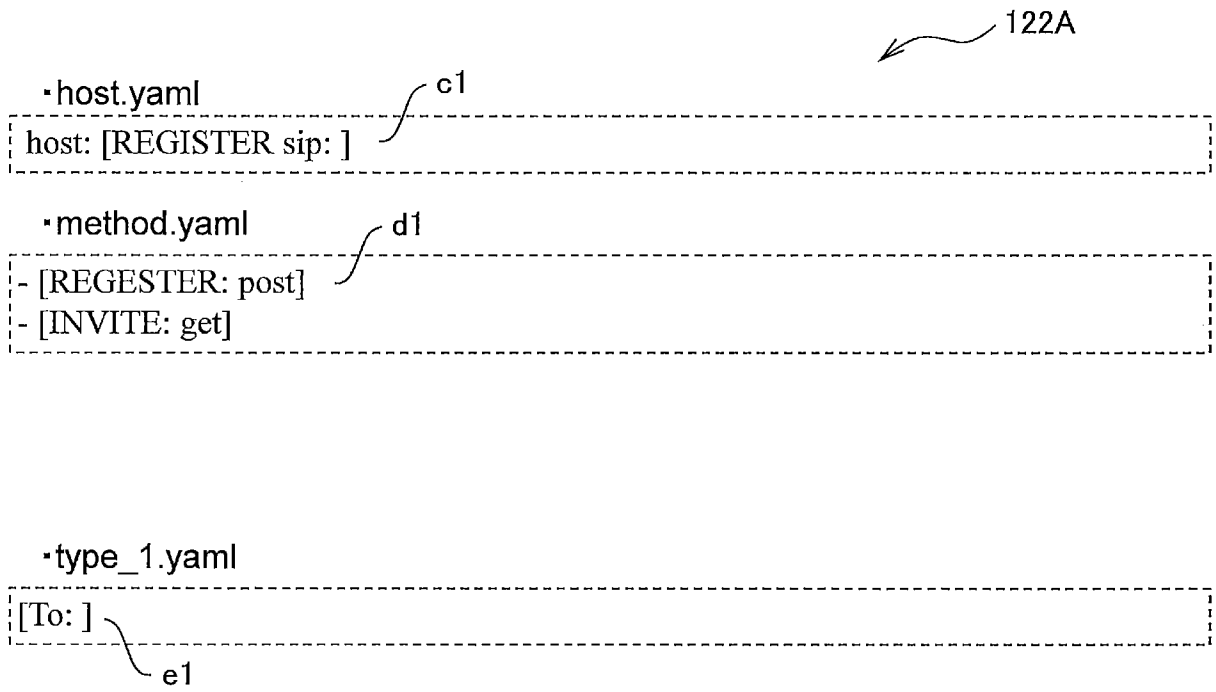
```
swagger: '2.0'
info:
  version: 1.0.0
  title: hogehoge
description: |

host: petstore-api.herokuapp.com
basePath: /pet
schemes:
  - http
consumes:
  - application/json
produces:
  - application/json
paths:
  /:

    post:
      parameters:
        - name: A
          in: body
          description: hogehogeschema:
            $ref: '#/definitions/A'
          required: true
      responses:
        200:
          description: hogehoge

definitions:
  A:
    type: object
    properties:
      name:
        type: string
      birthday:
        type: integer
        format: int32
```

[図7]

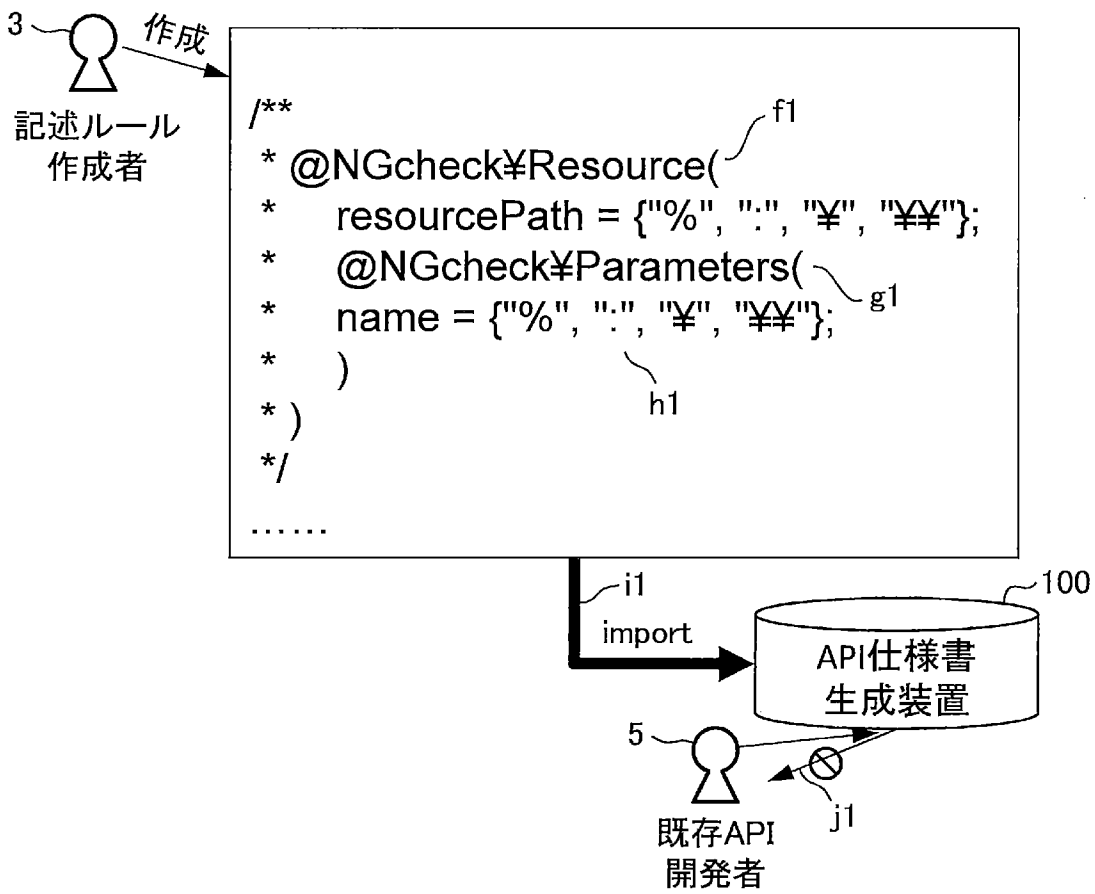
121A

```
swagger: '2.0'
info:
  version: 1.0.0
  title: hogehoge
  description: |hogehoge

  ! ! Conversion host:
  ! ! Conversion basePath:
  ! ! Conversion schemes:
  ! ! Conversion consumes:

  ! ! Conversion produces:
  ! ! Conversion Paths:

    ! ! Conversion method:
    parameters:
      - name: X
        ! ! Conversion in:
        description: hogehoge
        ! ! Conversion schema:
        required: true
      ! ! Conversion responses:
        ! ! Conversion response code:
        description: hogehoge

definitions:
  A:
    ! ! Conversion JavaScript Object:
    properties:
      from
        ! ! Conversion type_1:
        ! ! Conversion format_1
      to
        ! ! Conversion type_2:
        ! ! Conversion format_2:
```

[図8]


122A
・host.yaml
c1
host: [REGISTER sip:]
・method.yaml
d1
- [REGESTER: post]
- [INVITE: get]
・type_1.yaml
[To:]
e1

[図9]



[図10]

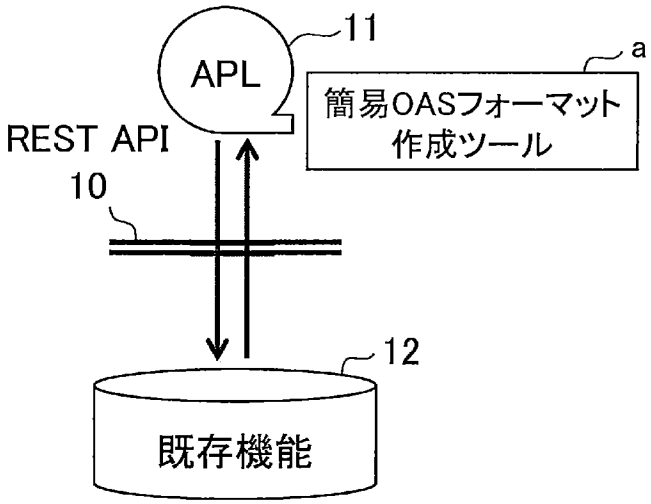
CRUD	メソッド名	用途
作成(Create)	POST	リソースを新規に「作成」
取得(Read)	GET	リソース情報の「読み出し」
更新(Update)	PUT	リソースの「更新」
削除(Delete)	DELETE	リソースの「削除」

[図11]

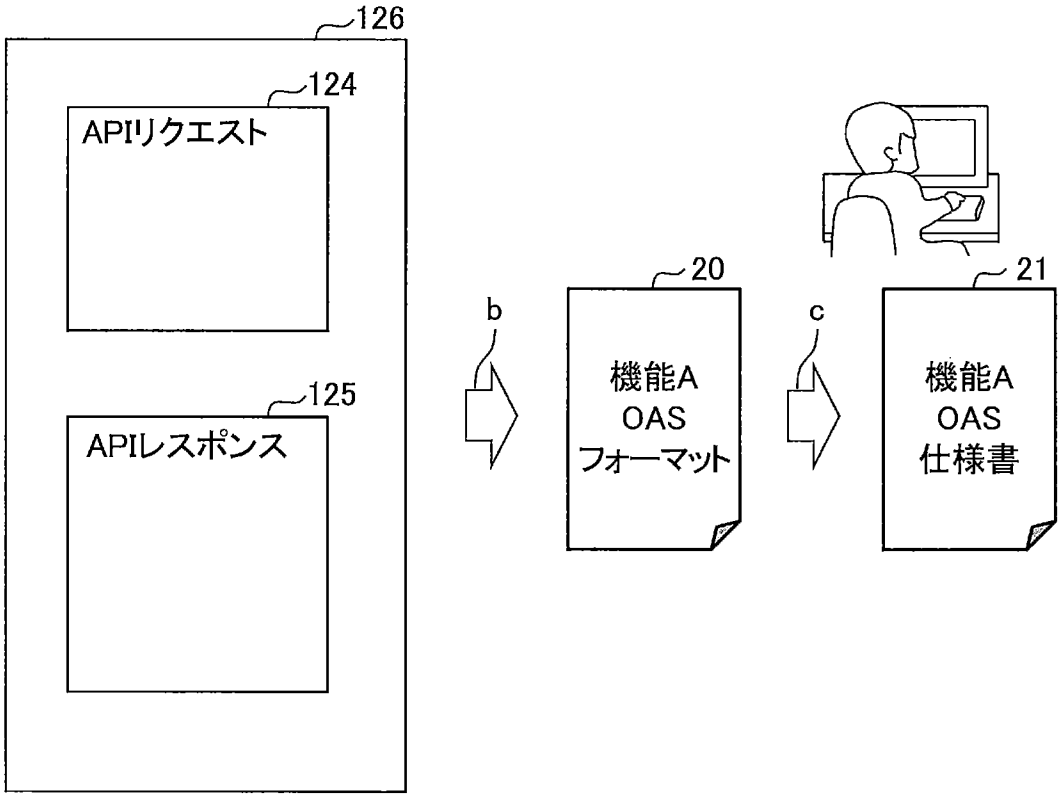
```
<?php
/**
 * @NGcheck¥Resource(
 *     resourcePath = {"%", ".", "¥", "¥¥"};
 *     @NGcheck¥Parameters(
 *         name = {"%", ".", "¥", "¥¥"};
 *     )
 * )
 * @NGcheck¥Api(                                    k1
 *     (description={“更新”, “create”, “追加”, “add“, “削除”, delete}) && (method="GET")) ,
 *                                                                    l1
 */
.....
```

[図12]

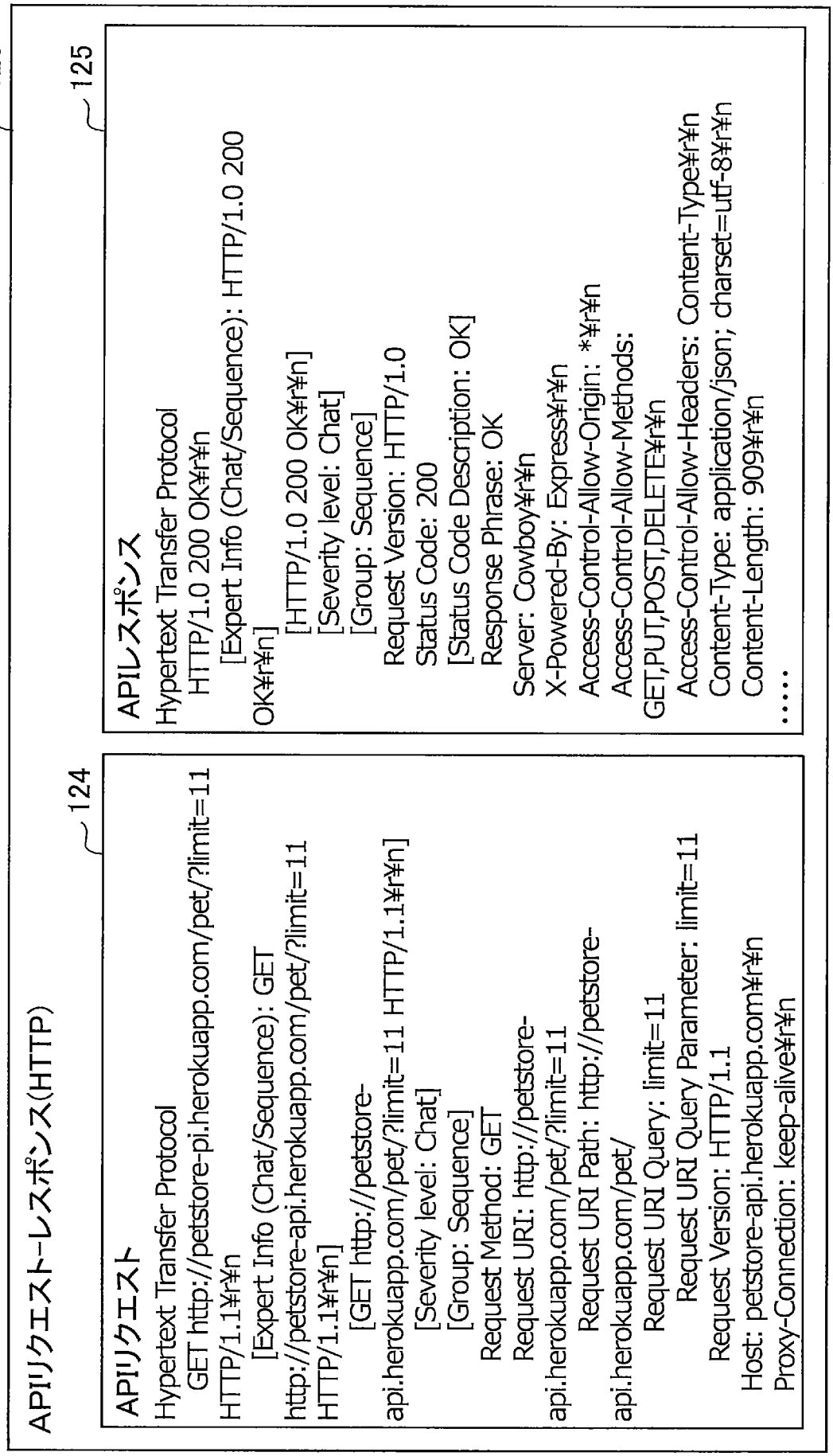
(a)


11
APL
a
簡易OASフォーマット
作成ツール
REST API
10
12
既存機能

(b)


126
124
APIリクエスト
125
APIレスポンス
b
20
機能A
OAS
フォーマット
c
21
機能A
OAS
仕様書

[図13]


126
APIリクエスト-レスポンス(HTTP)
124
APIリクエスト
Hypertext Transfer Protocol
GET http://petstore-pi.herokuapp.com/pet/?limit=11 HTTP/1.1¥r¥n
[Expert Info (Chat/Sequence): GET http://petstore-api.herokuapp.com/pet/?limit=11 HTTP/1.1¥r¥n]
[GET http://petstore-api.herokuapp.com/pet/?limit=11 HTTP/1.1¥r¥n]
[Severity level: Chat]
[Group: Sequence]
Request Method: GET
Request URI: http://petstore-api.herokuapp.com/pet/?limit=11
Request URI Path: http://petstore-api.herokuapp.com/pet/
Request URI Query: limit=11
Request URI Query Parameter: limit=11
Request Version: HTTP/1.1
Host: petstore-api.herokuapp.com¥r¥n
Proxy-Connection: keep-alive¥r¥n
125
APIレスポンス
Hypertext Transfer Protocol
HTTP/1.0 200 OK¥r¥n
[Expert Info (Chat/Sequence): HTTP/1.0 200 OK¥r¥n]
[HTTP/1.0 200 OK¥r¥n]
[Severity level: Chat]
[Group: Sequence]
Request Version: HTTP/1.0
Status Code: 200
[Status Code Description: OK]
Response Phrase: OK
Server: Cowboy¥r¥n
X-Powered-By: Express¥r¥n
Access-Control-Allow-Origin: *¥r¥n
Access-Control-Allow-Methods: GET,PUT,POST,DELETE¥r¥n
Access-Control-Allow-Headers: Content-Type¥r¥n
Content-Type: application/json; charset=utf-8¥r¥n
Content-Length: 909¥r¥n
.....

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2019/030228

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl. G06F8/73(2018.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl. G06F8/00-8/77, G06F9/54

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Published examined utility model applications of Japan	1922-1996
Published unexamined utility model applications of Japan	1971-2019
Registered utility model specifications of Japan	1996-2019
Published registered utility model applications of Japan	1994-2019

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2017/0012838 A1 (MICROSOFT TECHNOLOGY LICENSING, LLC) 12 January 2017, paragraphs [0028]-[0043], [0055]-[0069], fig. 1, 3 & WO 2017/007979 A1	1-5
A	US 9936005 B1 (KONG INC.) 03 April 2018, column 9, line 20 to column 13, line 18 (Family: none)	1-5

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search 06 September 2019 (06.09.2019)	Date of mailing of the international search report 17 September 2019 (17.09.2019)
Name and mailing address of the ISA/ Japan Patent Office 3-4-3, Kasumigaseki, Chiyoda-ku, Tokyo 100-8915, Japan	Authorized officer Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2019/030228

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 2018-55625 A (FUJITSU LTD.) 05 April 2018, paragraphs [0009]-[0011] & US 2018/0095810 A1, paragraphs [0010]-[0011]	1-5

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１９／０３０２２８

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F8/73(2018.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　G06F8/00-8/77, G06F9/54

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２０１９年
日本国実用新案登録公報	１９９６－２０１９年
日本国登録実用新案公報	１９９４－２０１９年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
A	US 2017/0012838 A1 (MICROSOFT TECHNOLOGY LICENSING, LLC) 2017.01.12, 段落[0028]-[0043],[0055]-[0069], 図 1,3 & WO 2017/007979 A1	１－５
A	US 9936005 B1 (KONG INC.) 2018.04.03, 第 9 欄第 20 行-第 13 欄第 18 行（ファミリーなし）	１－５

☑　Ｃ欄の続きにも文献が列挙されている。

☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
０６．０９．２０１９	１７．０９．２０１９

国際調査機関の名称及びあて先
日本国特許庁（ＩＳＡ／ＪＰ）
郵便番号１００－８９１５
東京都千代田区霞が関三丁目４番３号

特許庁審査官（権限のある職員）　５Ｂ　９３６７
多賀　実
電話番号　０３－３５８１－１１０１　内線　３５４５

Ｃ（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
Ａ	JP 2018-55625 A (富士通株式会社) 2018.04.05, 段落[0009]-[0011] & US 2018/0095810 A1, 段落[0010]-[0011]	１－５