Anotherware

OPC UA Specification

Part 4: Services

ver. 1.04.11

OPC UA 仕様書　4：サービス［日本語翻訳版］

INDEX

1. はじめに

本ドキュメントは、OPC UA 仕様書のサービスセットについての説明を抜粋したものです。詳しくは、OPC UA 仕様書を参照してください。原文のまま掲載している箇所もありますが、ご了承ください。

2. 概要

2.1. サービスセット model

ここでは、OPC UA サービスについて記載しています。
ただし OPC UA サービス定義は抽象化されたものであり、具体的な実装についての記載はしていません。
OPC UA サービスをウェブサービスとしての実装する場合には、ウェブサービスとウェブサービスの運用にも適応する必要があります。
これらのサービスはサービスセットという論理的なグループで構成されており、各サービスセットは関連するサービスに関して定義しています。
図 1 に示す Discovery サービスのサービスセットは、クライアントがサーバで実装されているエンドポイントを検出し、それらの各エンドポイントのセキュリティ構成を読み取ることができるサービスを定義しています。

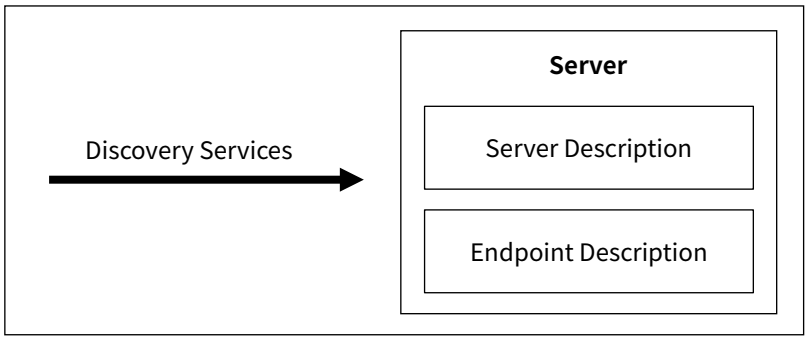

図 1 - Discovery Service Set

図 2 に示すセキュアチャネルサービスのサービスセットでは、クライアントが通信チャネルを確立して、サーバと送受信するメッセージの機密性と完全性を確実にするサービスを定義しています。

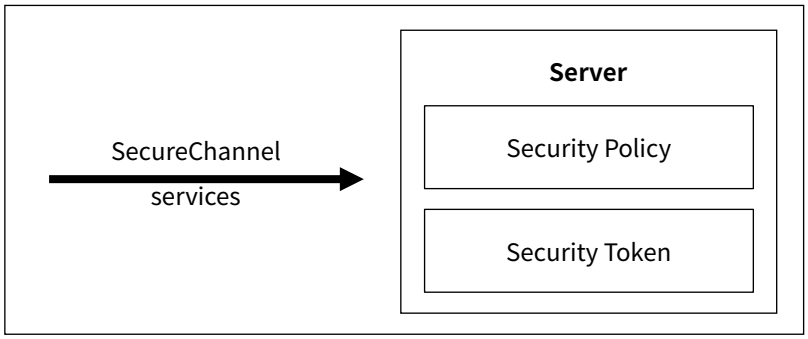

図 2 - SecureChannel Service Set

図 3 に示す Session サービスのサービスセットでは、クライアントのユーザ認証やセッションを管理するためのサービスを定義しています。

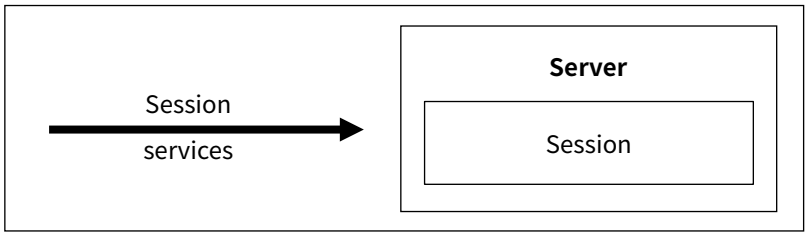

図 3 - Session Service Set

図 4 に示す NodeManagement サービスのサービスセットでは、クライアントがアドレス空間にノードを追加、変更および削除ができるようにするサービスを定義しています。

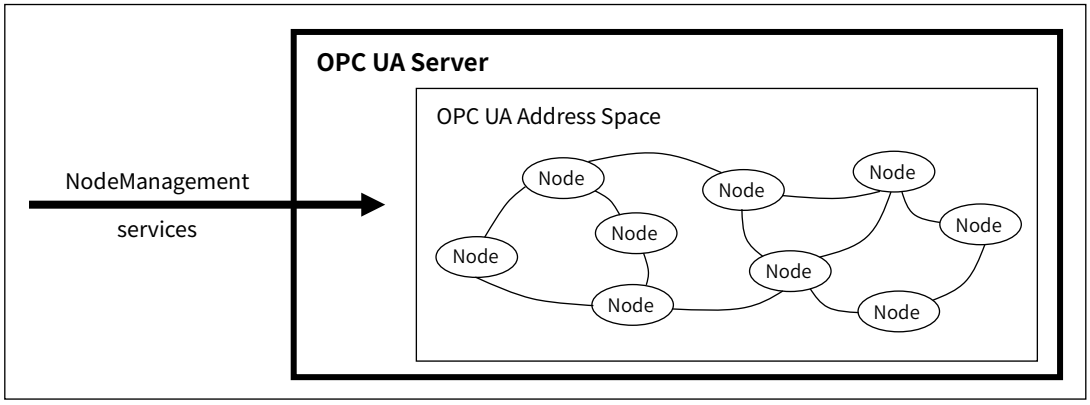

図 4 - NodeManagement Service Set

図 5 に示す View サービスのサービスセットは、クライアントがアドレス空間や View というアドレス空間のサブセットを参照できるようにするサービスを定義しています。 Query サービスセットを使用すると、クライアントはアドレス空間や View から、データのサブセットを取得することができます。

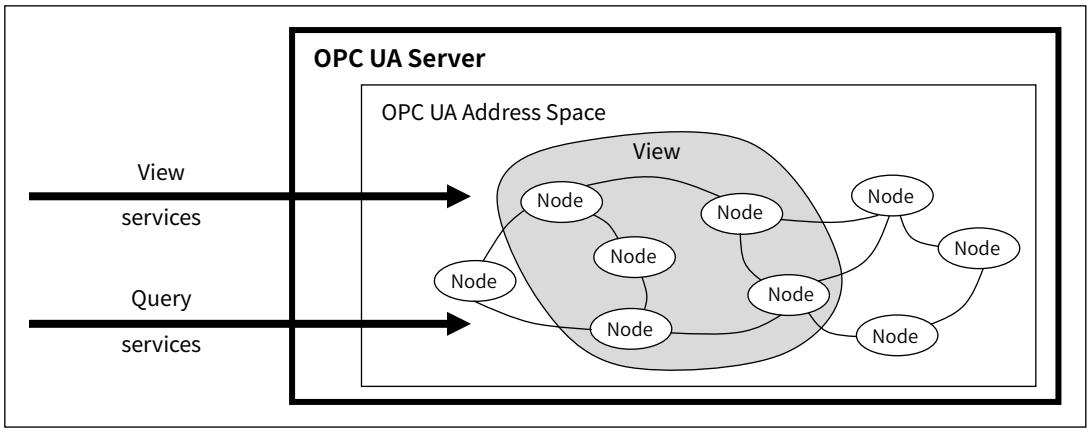

図 5 - View Service Set

図 6 に示す Attribute サービスのサービスセットは、クライアントが履歴を含むノードの属性を読み書きできるサービスを定義しています。 履歴などの属性値はモデル化されているため、Attribute サービスセットの各サービスによってクライアントは属性値を読み書きすることができます。

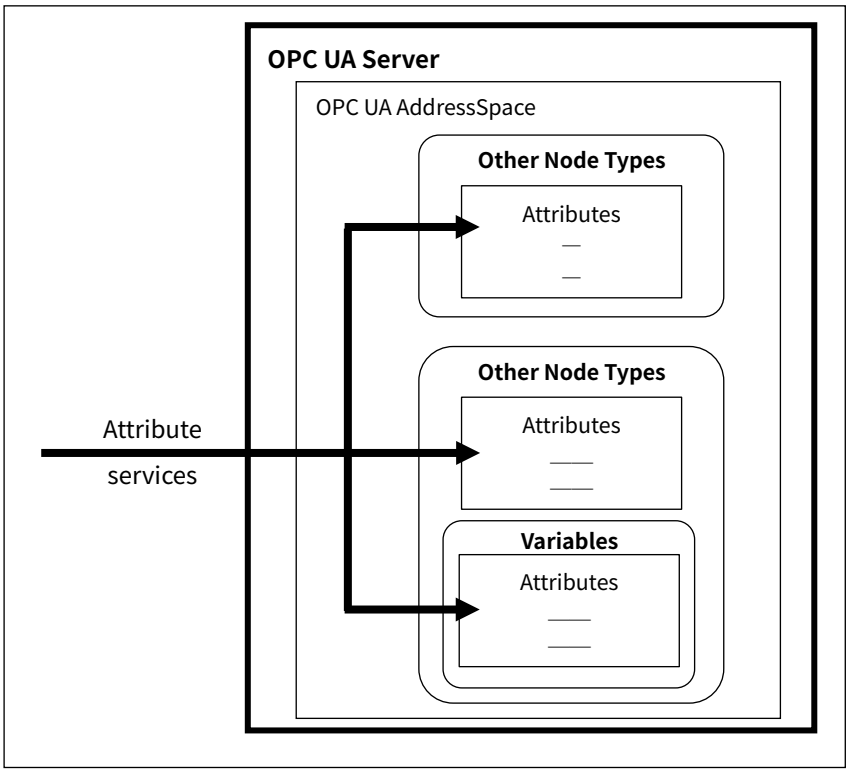

図 6 - Attribute Service Set

図 7 に示す Method サービスのサービスセットは、クライアントがメソッドを呼び出すことを可能にするサービスを定義しています。 メソッドが呼び出されると、完了するまで実行されます。メソッドは、各メソッドで指定されている入力パラメータで呼び出されます。それらは、出力パラメータを返すこともあります。

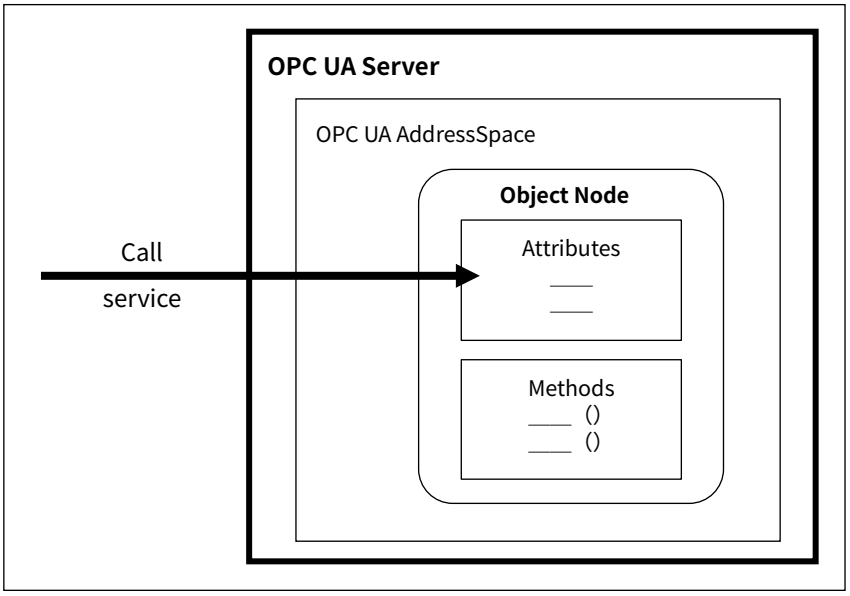

図 7 - Method Service Set

図 8 に示す MonitoredItem サービスと Subscription サービスセットは、OPC UA アドレス空間のノードに Subscribe(定期通知)するために同時に使用されます。

MonitoredItem サービスセットは、値やオブジェクトを監視するために使用される MonitoredItem をクライアントが作成、変更、および削除できるようにするサービスを定義しています。

Subscription サービスセットは、クライアントがサブスクリプションの作成、変更、および削除を可能にするサービスを定義しています。

監視対象に変化があった場合などの通知は、MonitoredItem サービスによってサブスクリプションごとにクライアントに転送するためにキューに入れられます。

サブスクリプションは、MonitoredItems によってキューに入れられた通知をクライアントに送信します。

Subscription サービスでは、通信障害などによるサブスクリプションによるエラー発生時のリカバリも提供します。

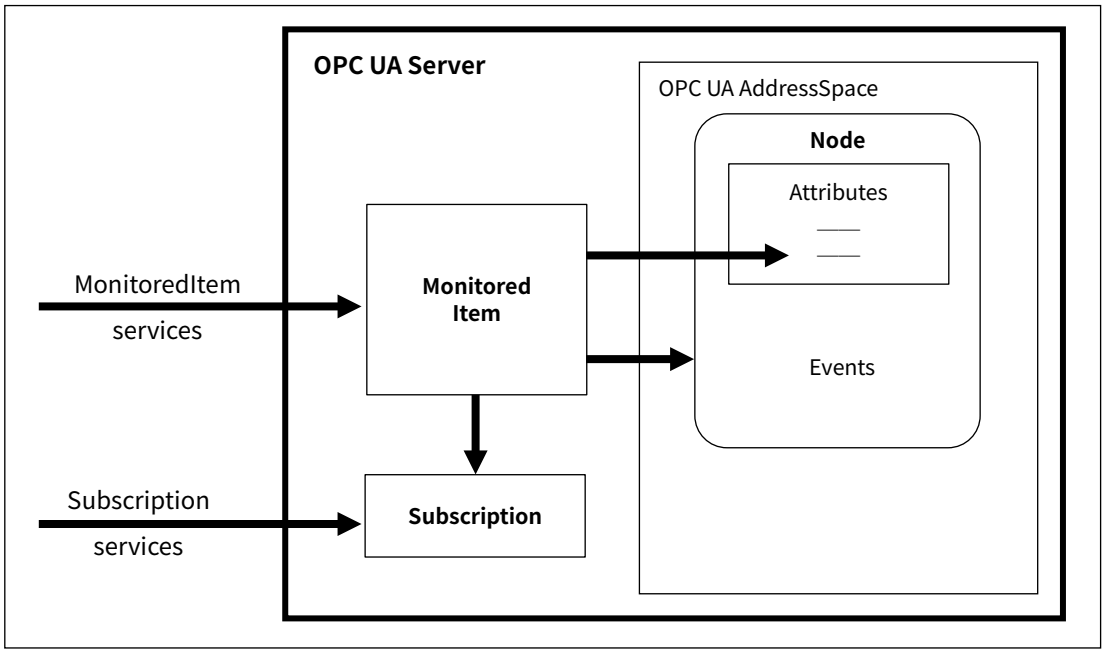

図 8 - MonitoredItem and Subscription Service Sets

2.2. Request と Response の仕組み

クライアントがサーバにリクエストを送信すると、サーバは２つのステップでリクエストを処理します。
最初のステップではリクエストをデコードしてサービスを特定し、次のステップで特定されたサービスの各操作を実行します。
リクエストの各操作では出力データとその結果を返し、レスポンスには各操作毎にそれらを生成し送信します。
これらの操作を実現するために、クライアントおよびサーバは OPC UA 通信スタックの API を使用して、リクエストやレスポンスの生成及び解析、さらにリクエスト操作を実行します。しかし、それらを実行する前に、各サービスの実装では、リクエストやレスポンス内のパラメータの正誤チェックを行わなければなりません。

3. サービスセット

3.1. 概要

この章では、OPC UA サービスセットとそのサービスを定義します。

サーバがサービスセットまたはサービスセット内のサービスをサポートしているかどうかは、そのプロファイルによって定義されます。プロファイルについてはここでは省略します。

3.2. サービスの Request ヘッダと Response ヘッダ

各サービスの要求には RequestHeader があり、各サービス応答には ResponseHeader があります。
RequestHeader は構造体定義されており、いくつかの共通要求パラメータを含んでいます。
ResponseHeader は構造体定義されており、一般的な応答パラメータを含みます。

3.3. サービス結果

サービス結果は、OPC UA 応答の 2 つのレベルで返されます。
1 つはサービス呼び出しのステータスを示し、もう 1 つはサービスによって要求された各操作のステータスを示します。

サービス呼び出しのステータスは、ResponseHeader に含まれる serviceResult によって表されます。
このパラメータを返すメカニズムは、Service レスポンスを伝送するために使用される通信テクノロジに固有の定義となります。

要求内の個々の操作のステータスは、個々のステータスコードによって表されます。

次のケースでは、これらのパラメータの使用を定義します。

a) サービス自体が失敗した場合、serviceResult に不正なコードが返されます。
この場合、ServiceFault が返されます。
b) サービスが完全にまたは部分的に成功した場合、正常コードが serviceResult に返されます。この場合は、他の応答パラメータが返されます。クライアントは、応答パラメータ、特に各操作に関連するすべてのステータスコードを常にチェックしなければなりません。
これらのステータスコードは、サービスコールで要求された 1 つ以上の操作の結果が不良または不確実であることを示している可能性があります。要求に操作の配列を持つすべてのサービスは、配列が空の場合には serviceResult に不正なコードを返すことになります。

サービスには各種特定のステータスコードを定義し、サーバはサービスに記載されている特定のステータスコードを使用します。
クライアントは、これらのサービス固有のステータスコードを処理できる必要があります。さらに、クライアントは共通

のステータスコードも待ち受ける必要があります。

セッションまたはセキュアチャネルの作成に使用されたアプリケーションまたはユーザの資格情報が改竄されたことをサーバが検出した場合、サーバはこれらの資格情報を使用するすべてのセッションとチャネルを直ちに終了する必要があります。この場合、サービス結果コードは Bad_IdentityTokenRejected または Bad_CertificateUntrusted のいずれかにする必要があります。

構文エラーまたはメッセージに含まれるデータが受信側でサポートされている範囲を超えているため、メッセージ解析が失敗する可能性があります。この場合、メッセージは受信者によって拒絶されます。受信者がサーバの場合、サービス結果コードは Bad_DecodingError または Bad_EncodingLimitsExceeded を持つ ServiceFault が返されます。受信者がクライアントの場合、通信スタックはこれらのエラーをクライアントアプリケーションに通知する必要があります。

多くのアプリケーションは、これらのメッセージに含まれるメッセージやデータ要素のサイズに制限を設けます。たとえば、ある長さより長い文字列値を含む要求を拒否することがあります。これらの制限は、通常管理者によって設定され、クライアントとサーバ間のすべての接続に適用されます。

サーバから Bad_EncodingLimitsExceeded 問題を受け取ったクライアントは、要求を再調整する必要があります。管理者は、この問題を持つサーバから応答を受け取った場合、クライアントの制限を増やす必要があります。

場合によっては、解析エラーが致命的であるため、問題を報告できないことがあります。例えば、着信メッセージは、受信機のバッファ容量を超える可能性があります。このような場合、これらのエラーは、セキュアチャネルの再確立が必要な通信障害として扱われる可能性があります。

クライアントとサーバは、CreateSession サービスでメッセージサイズ制限を交換することで、致命的なエラーの可能性を減らします。これにより、どちらの当事者も通信障害の原因となるメッセージの送信を避けることができます。シリアライズされた応答メッセージがクライアントによって指定されたメッセージサイズを超える場合、サーバは Bad_ResponseTooLarge を返します。同様に、クライアント通信スタックは、サーバの制限を超えるメッセージを送信する前に、アプリケーションに Bad_RequestTooLarge エラーを報告する必要があります。

メッセージサイズの制限はメッセージ本文にのみ適用され、ヘッダやセキュリティには適用されないことに注意してください。これは、指定された最大値より小さいメッセージ本文が依然として致命的なエラーを引き起こす可能性があることを意味します。

3.4. Discovery サービスセット

3.4.1.概要

このサービスセットは、サーバによって実装されたエンドポイントを検出し、それらのエンドポイントのセキュリティ構成を読み取るために使用されるサービスを定義します。Discovery サービスは、個々のサーバと専用の Discovery サーバによって実装されます。

すべてのサーバは、クライアントがセッションを確立せずにアクセスできる Discovery エンドポイントを持つものとします。このエンドポイントは、クライアントがセキュアチャネルの確立に使用するのと同じセッションエンドポイントでなくても構いません。クライアントは、Discovery エンドポイントで GetEndpoints サービスを呼び出してセキュアチャネルを確立するために必要なセキュリティ情報を読み取ります。

さらに、サーバは、RegisterServer サービスを使用して、明示的に Discovery サーバに登録することができます。
クライアントは、Discovery サーバ上の FindServers サービスを呼び出すことによって、登録済みのサーバを検出できます。

FindServers を使用した検出プロセスを図 9 に示します。
FindServers および GetEndpoints のセキュアチャネル(MessageSecurityMode NONE を使用)の確立は、簡易記載のため図から省略されています。

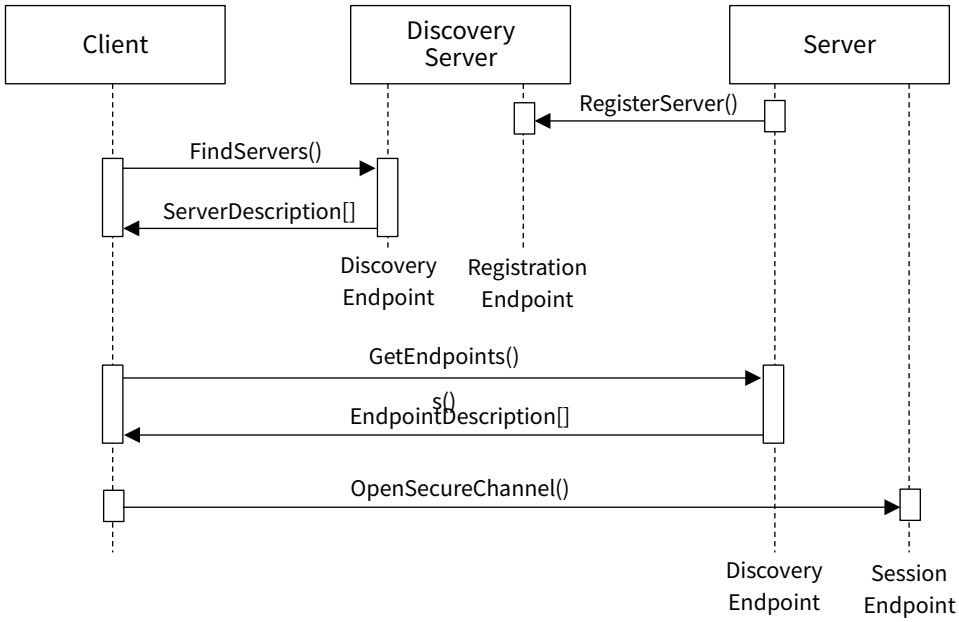

図 9 - 検出プロセス

Discovery エンドポイントの URL は、クライアントが Discovery エンドポイントに接続する必要があるすべての情報を提供するものとします。

クライアントがエンドポイントを取得すると、クライアントはこの情報を保存し、それを使用して Discovery プロセスを経ずにサーバに直接接続できます。クライアントが接続できないと判明した場合、サーバ構成が変更され、クライアントが再度 Discovery プロセスを実行する必要があります。

Discovery エンドポイントはメッセージのセキュリティを必要としませんが、トランスポート層のセキュリティが必要な場合があります。システムの運用においては、管理者はセキュリティ上の理由で検出を無効にし、クライアントはキャッシュされた EndpointDescriptions に依存して動作します。

Discovery サービスを無効にしたシステムをサポートするには、管理者はサーバへの接続に使用される EndpointDescriptions を手動で更新する必要があります。
サーバの Discovery エンドポイントは、管理者により無効化されます。

Discovery エンドポイントから返された情報を使用するときは、セキュリティがないため、クライアントは注意が必要です。クライアントは、Discovery エンドポイントから返された情報と CreateSession 応答で返された情報を比較することによってこれを行います。
クライアントは、以下を確認しなければなりません。

a) サーバ証明書で指定された ApplicationUri が、EndpointDescription で提供される ApplicationUri と同じであること。
b) CreateSession 応答で返されるサーバ証明書は、セキュアチャネルの作成に使用された証明書と同じであるこおと。
c) Discovery エンドポイントから返された EndpointDescriptions は、CreateSession レスポンスで返された EndpointDescriptions と同じであること。

クライアントが上記の要件の 1 つが満たされていないことを検出した場合、クライアントはセキュアチャネルを閉じてエラーを報告します。

クライアントは、サーバ証明書に指定された HostName が、CreateSession によって返された EndpointDescription で提供される endpointUrl に含まれる HostName と同じであることを確認しなければなりません。差異がある場合、クライアントは差異を通知し、セキュアチャネルを閉じることができます。サーバは、MyHost および MyHost.local のようなすべての可能なホスト名をサーバ証明書に追加しなければなりません。これには、ホストの IP アドレスまたはサーバに接続するために使用される NAT ルーターによって公開される HostName が含まれます。

3.4.2.FindServers

3.4.2.1. 説明

このサービスは、サーバまたは Discovery サーバに認識されているサーバを返します。

クライアントは、フィルタ基準を指定することによって返される結果の数を制限することができます。Discovery サーバは、クライアントが指定した基準に一致するサーバがない場合、空のリストを返します。

すべてのサーバは、本サービスをサポートする Discovery エンドポイントを提供します。サーバは、それ自体を記述するレコードを常に返しますが、場合によっては複数のレコードが返されることもあります。ゲートウェイサーバは、ゲートウェイサーバに通常の OPC UA サーバとしてアクセスできるようにするレコード(オプション)にアクセスを提供する各サーバのレコードを返すものとします。非透過冗長サーバは、冗長サーバセット内の各サーバのレコードを提供するものとします。

すべてのサーバは、ServerUri と呼ばれるグローバルに一意の識別子を持ちます。この識別子は、完全修飾ドメイン名である必要があります。ただし、グローバル一意性を保証する GUID または類似の構造である可能性があります。本サービスから返される ServerUri は、ServerArray プロパティのインデックス 0 に表示されるものと同じ値になります。ServerUri は、ApplicationDescription の applicationUri フィールドとして返されます。

すべてのサーバには、必ずしもグローバルに一意であるとは限りません。この識別子は複数のロケールで使用できます。

サーバは複数のホスト名を持つことがあります。このため、クライアントはエンドポイントに接続するために使用した URL を本サービスに渡すものとします。本サービスの実装は、この情報を使用して、提供された URL を介してクライアントがアクセス可能な応答を返すものとします。

本サービスはメッセージセキュリティを必要としませんが、トランスポート層のセキュリティが必要な場合があります。

一部のサーバは、ゲートウェイサーバ経由でアクセスすることができ、その ApplicationDescription に gatewayServerUri に指定された値を持たなければなりません。
ApplicationDescription で提供される discoveryUrl は、ゲートウェイサーバに属しているものとします。
一部の検出サーバは、複数のパス経由でそのサーバにアクセスできる場合、同じサーバに対して複数のレコードを返すことがあります。

このサービスはセキュリティなしで使用できるため、サービス拒否(DOS)攻撃に対して脆弱です。
サーバは、このサービスの応答を送信するために必要な処理量を最小限に抑える必要があります。
これは、事前に結果を準備することで実現できます。
また、サーバは、トラフィックが多い状況で要求の処理を開始する前に短いディレイを追加する必要があります。

3.4.2.2. パラメータ

表 1 は、サービスのパラメータを定義しています。

表 1 - FindServers サービスのパラメータ

Name	Type	Description
Request		
requestHeader	RequestHeader	共通の要求パラメータ。 authenticationToken は常に null です。 authenticationToken は、提供されている場合は無視されます。

		RequestHeader 型は 7.28 で定義されている。
endpointUrl	String	クライアントが Discovery エンドポイントにアクセスするために使用したネットワークアドレス。 サーバはこの情報を診断に使用し、応答で返す URL を決定します。 URL 内の HostName が認識されない場合、サーバは適切なデフォルト URL を返す必要があります。
localeIds []	LocaleId	使用するロケールのリスト。 サーバは、指定されたロケールの 1 つを使用して 7.1 で定義された ApplicationDescription の applicationName を戻す必要があります。 サーバが要求されたロケールの複数をサポートしている場合、サーバはこのリストの最初に表示されるロケールを使用します。 サーバが要求されたロケールをサポートしていない場合は、適切なデフォルトロケールが選択されます。 このリストが空の場合、サーバは適切なデフォルトロケールを選択します。
serverUris []	String	戻すサーバのリスト。 リストが空の場合は、すべての既知のサーバが戻されます。 serverUri は、7.1 で定義された ApplicationDescription の applicationUri と一致します。
Response		
responseHeader	ResponseHeader	共通の応答パラメータ（7.29 ResponseHeader 型定義を参照）。
servers []	ApplicationDescription	要求で指定された基準を満たすサーバのリスト。 このリストは、サーバが基準を満たさない場合は空です。 ApplicationDescription タイプは 7.1 で定義されています。

3.4.2.3. サービス結果

Common StatusCodes は表 176 で定義されています。

3.4.3.FindServersOnNetwork

3.4.3.1. 説明

このサービスは、Discovery サーバに認識されているサーバを返します。FindServers と異なり、このサービスは Discovery サーバによってのみ実装されます。

クライアントは、フィルタ基準を指定することによって返される結果の数を制限することができます。サーバがクライアントによって指定された基準に一致しない場合、空のリストが返されます。

本サービスはメッセージセキュリティを必要としませんが、トランスポート層のセキュリティが必要な場合があります。

Discovery サーバは、キャッシュ内のレコードを作成または更新するたびに、増加する識別子をレコードに割り当てます。
これにより、クライアントは、FindServersOnNetwork への最後の呼び出しで受信した最後のレコードの識別子を指定することにより、バッチでレコードを要求できます。
これをサポートするために、Discovery サーバは、最も低いレコード ID から始まるレコードを数値順に返します。
また、Discovery サーバは、Discovery サーバの再起動のためにカウンタがリセットされた最後の時間を返します。
クライアントでこの時間が最後にクライアントがサービスを呼び出した時刻より新しいことを検出した場合、クライアントは startRecordId を 0 にしてサービスを再度呼び出します。

このサービスはセキュリティなしで使用できるため、サービス拒否(DOS)攻撃の脆弱性があります。
サーバは、このサービスの応答を送信するために必要な処理量を最小限に抑える必要があります。
これは、事前に結果を準備することで実現できます。

3.4.3.2. パラメータ

表 4 は、サービスのパラメータを定義しています。

表 2 - FindServersOnNetwork サービスのパラメータ

Name	Type	Description
Request		
requestHeader	RequestHeader	共通の要求パラメータ。 authenticationToken は常に null です。 authenticationToken は、提供されている場合は無視されます。 RequestHeader 型は 7.28 で定義されています。
startingRecordId	Counter	この番号より大きな識別子を持つレコードだけが返されます。 キャッシュ内の最初のレコードで開始するには 0 を指定します。
maxRecordsToReturn	UInt32	応答で返すレコードの最大数。 0 は制限がないことを示します。
serverCapabilityFilter[]	String	サーバ機能フィルタのリスト 許可されたサーバ機能のセットは、パート 12 で定義されています。 指定されたすべてのサーバ機能を持つレコードのみが返されます。 比較では大文字と小文字を区別しません。 このリストが空の場合、フィルタリングは実行されません。
Response		
responseHeader	ResponseHeader	共通の応答パラメータ（7.29 ResponseHeader 型定義を参照）。
lastCounterResetTime	UtcTime	カウンタがリセットされた最後の時間。
servers[]	ServerOnNetwork	要求で指定された基準を満たす DNS サービスレコードのリスト。 サーバが基準を満たさない場合、このリストは空です。

recordId	UInt32	レコードの一意の識別子。 これは、その後の FindServersOnNetwork の呼び出しでサーバの次のバッチをフェッチするために使用できます。
serverName	String	mDNS アナウンスメントで指定されたサーバの名前（パート 12 を参照）。 これは、サーバの ApplicationName と同じでもかまいません。
discoveryUrl	String	検出エンドポイントの URL。
serverCapabilities	String[]	サーバによってサポートされる一連のサーバ機能。 許可されたサーバ機能のセットは、パート 12 で定義されています。

3.4.3.3. サービス結果

Common StatusCodes は表 176 で定義されています。

3.4.4.GetEndpoints

3.4.4.1. 説明

このサービスは、サーバがサポートするエンドポイントと、セキュアチャネルとセッションを確立するために必要なすべての構成情報を返します。

本サービスはメッセージセキュリティを必要としませんが、トランスポート層のセキュリティが必要な場合があります。

クライアントは、LocaleIds とトランスポートプロファイル URI に基づいてフィルタ条件を指定することによって返される結果の数を制限することができます。エンドポイントがクライアントによって指定された基準と一致しない場合、サーバは空のリストを返します。

サーバは、同じエンドポイントに対して複数のセキュリティ構成をサポートする場合があります。この場合、サーバは利用可能な構成ごとに別々の EndpointDescription レコードを返すものとします。クライアントは、たとえ物理 URL が同じであっても、これらの設定のそれぞれを別個のエンドポイントとして扱う必要があります。

エンドポイントのセキュリティ設定には、次の 4 つのコンポーネントがあります。

- サーバアプリケーションインスタンス証明書
- メッセージセキュリティモード
- セキュリティポリシー
- サポートされているユーザ ID トークン

ApplicationInstanceCertificate は、Open セキュアチャネル要求を保護するために使用されます。MessageSecurityMode と SecurityPolicy は、セキュアチャネルを介して送信されたメッセージを保護する方法をクライアントに通知します。

UserIdentityTokens は、クライアントに、ActivateSession 要求でどのタイプのユーザ資格情報をサーバに渡すかを指示します。

securityPolicyUri が NONE であり、UserTokenPolicies のいずれも暗号化を必要としない場合、クライアントは ApplicationInstanceCertificate を無視するものとします。

各 EndpointDescription は、エンドポイントがサポートする伝送プロファイルの URI も指定します。伝送プロファイルは、メッセージエンコーディングフォーマットやプロトコルバージョンなどの情報を指定します。クライアントは、サーバがサポートしているプロファイルの完全なリストを発見したい場合は、サーバのソフトウェア証明書を取り出す必要があります。

メッセージは、ネットワークを介して送信される前にメッセージに標準の暗号化アルゴリズムを適用することによって保護されます。使用される正確なアルゴリズムのセットは、エンドポイントの SecurityPolicy によって異なります。アプリケーションには、サポートする SecurityPolicies が組み込まれていることが期待されます。その結果、SecurityPolicy のプロファイル URI のみが EndpointDescription に指定されます。クライアントは、認識した SecurityPolicy をサポートしていないエンドポイントに接続できません。

EndpointDescription は、メッセージセキュリティモードが NONE であることを指定することができます。侵入のリスクが非常に小さい物理的に隔離されたネットワーク上でアプリケーションが通信している場合を除き、この設定は推奨されません。メッセージのセキュリティが NONE の場合、クライアントは意図的にまたは偶然に他のクライアントによって作成されたセッションを乗っ取ることができます。

サーバは複数のホスト名を持つことがあります。このため、クライアントはエンドポイントに接続するために使用した URL を本サービスに渡すものとします。本サービスの実装は、この情報を使用して、提供された URL を介してクライアントがアクセス可能な応答を返すものとします。

このサービスはセキュリティなしで使用できるため、サービス拒否(DOS)攻撃に対して脆弱です。
サーバは、このサービスの応答を送信するために必要な処理量を最小限に抑える必要があります。
これは、事前に結果を準備することで実現できます。また、サーバは、トラフィックが多い状況で要求の処理を開始する前に短いディレイを追加する必要があります。

応答に返される EndpointDescriptions の中には、別のサーバにアクセスするために使用できるゲートウェイサーバのエンドポイント情報を指定するものがあります。このような状況では、gatewayServerUri が EndpointDescription で指定され、証明書の検証に使用されるすべてのセキュリティチェックでは、serverUri の代わりに gatewayServerUri を使用します。

ゲートウェイを介してサーバに接続するには、クライアントは最初にゲートウェイサーバとセキュアチャネルを確立します。クライアントは CreateSession サービスを呼び出し、EndpointDescription で指定された serverUri をゲートウェイサ

ーバに渡します。ゲートウェイサーバは、クライアントのために基底のサーバに接続しなければなりません。図 10 に、ゲートウェイサーバ経由でサーバに接続するプロセスを示します。

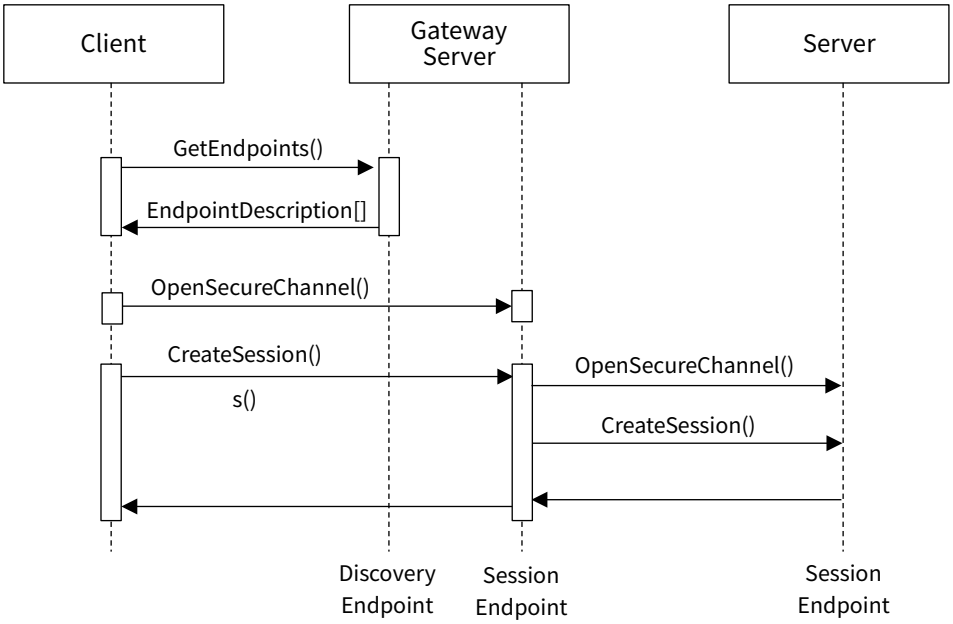

図 10 - ゲートウェイサーバの使用

3.4.4.2. パラメータ

表 3 は、サービスのパラメータを定義しています。

表 3 - GetEndpoints サービスのパラメータ

Name	Type	Description
Request		
requestHeader	RequestHeader	共通の要求パラメータ。 authenticationToken は常に null です。 authenticationToken が提供されている場合は無視されます。 RequestHeader 型は 7.28 で定義されています。
endpointUrl	String	クライアントが Discovery エンドポイントにアクセスするために使用したネットワークアドレス。 サーバはこの情報を診断に使用し、応答で返す URL を決定します。 URL 内の HostName が認識されない場合、サーバは適切なデフォルト URL を返す必要があります。
localeIds []	LocaleId	使用するロケールのリスト。 人間が読める文字列を返すときに使用するロケールを指定します。 このパラメータについては、5.4.2.2 で説明します。。
profileUris []	String	返されたエンドポイントがサポートするトランスポートプロファイルのリスト。 7 章には、トランスポートプロファイルの URI が定義されています。

本資料の続き（全 119 ページ）をご覧になりたい方は、
株式会社アナザーウェアまでお問合せください。

CONTACT

OPC UA の開発・サポートのご相談も以下のメールから受け付けております。

opc-ua-itron-toolkit@another-ware.co.jp

担当：山浦・鬼澤

株式会社アナザーウェア

本社

〒221-0835　横浜市神奈川区鶴屋町 2-21-8 第 1 安田ビル 6F

東京オフィス

〒153-0064　東京都目黒区下目黒 3-7-2 小西ビル 3F

www.another-ware.co.jp