組込み分野のための
UML モデル解説書

機能編 F001

認証

本書は、UML モデルカタログに含まれる「認証」のモデルの詳細を記述したものです。モデリングの初心者には教科書や参考書として、モデリングのベテランの方々にはモデルのヒントとして、ぜひともお手元に置いて活用してください。

UMTP 組込み
モデリング部会

2016.5.15 更新

Copyright(c)2010-2016 consortium for UML based Modeling Technologies Promotion All Rights Reserved

UMTP は特定非営利活動法人 UML モデリング推進協議会の登録商標です。その他、本書に記載されている会社名、商品名などは、一般に各社の商標または登録商標です。

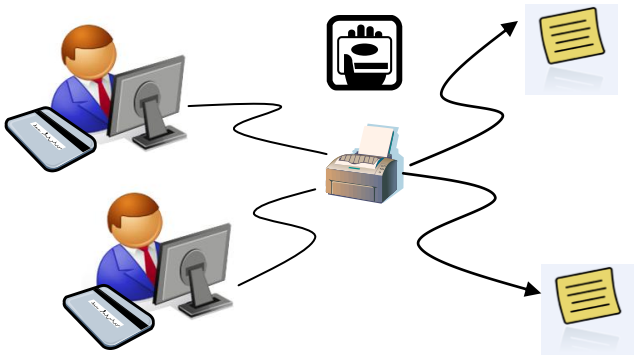
目　次

要求仕様 4
モデル一覧 10
機能に着目したモデル 11
　分析モデル 11
　PIM 設計モデル 17
エンティティに着目したモデル 18
　分析モデル 18
　PIM 設計モデル 24
状態に着目したモデル 29
　分析モデル 29
　PIM 設計モデル 32

要求仕様

認証例

ユーザ A とユーザ B がいます。二人はそれぞれ固有の ID カードを持っています。この 2 人が 2 台の PC からそれぞれの電子データを 1 台のプリンタに印刷指示を出します。ユーザ A がプリンタに ID カードをかざすと、ユーザ A が印刷指示した電子データが呼び出され、プリントが始まります。ユーザ A の電子データの印刷処理終了後、ユーザ B が同様に ID カードをプリンタにかざすと、ユーザ B の電子データが呼び出されプリントが始まります。ユーザ A,B の使用日時や電子データのタイトル、プリント枚数など使用記録がプリンタ保存されます。



このようなユーザごとに適切なサービスを提供できる仕組みを【認証】と呼称します。

この【認証】では、ユーザを識別し、ユーザごとに適切なサービスを提供し、機器の使用記録を取ります。この一連の機能の設計モデルを提供します。

ユースケース

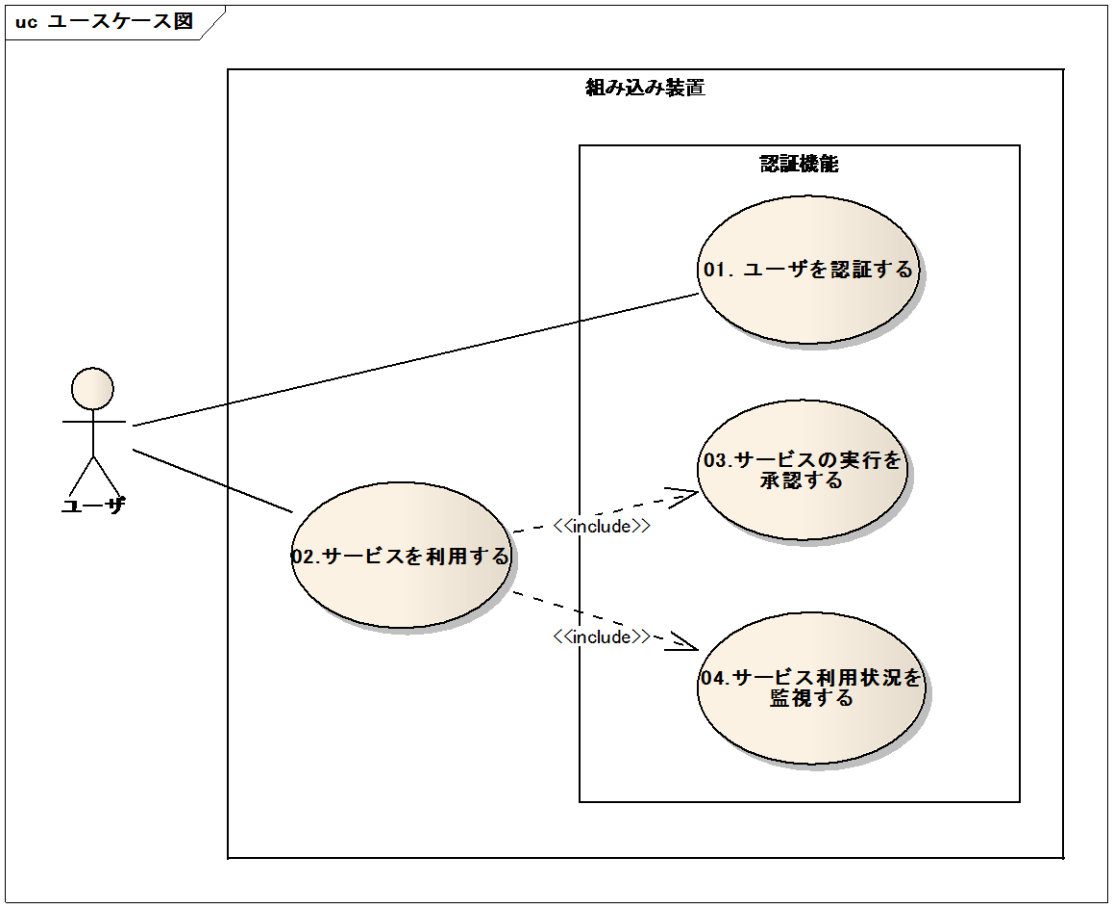

uc ユースケース図
組み込み装置
認証機能
01. ユーザを認証する
03.サービスの実行を承認する
02.サービスを利用する
<<include>>
<<include>>
04.サービス利用状況を監視する
ユーザ

図1

ユースケース記述

＜UC01 名：ユーザを認証する＞

■概要

ユーザが正当なユーザであるかどうかを判断する

■アクター

ユーザ

■事前条件

なし

■事後条件

ユーザに認証結果が渡されていること

■メインフロー

1 .アクターは、システムに対し、アクセス許可を要求する
2. システムは、アクターが正当なユーザかどうか判断する
3. システムは、アクターに対し、認証結果を渡す
4. UC を終了する

■課題や T．B．D 項目

なし

■備考

なし

＜UC02 名：サービスを利用する＞

■概要

ユーザが組み込み装置で提供されているサービスを利用する

■アクター

ユーザ

■事前条件

ユーザが認証済みである

■事後条件

ユーザに認証結果が渡されている

■メインフロー

1 .アクターは、システムに対し、サービスの利用を要求する
2. システムは、認証機能に対し、「サービスの実行を承認する」ユースケースを実行する
3. システムは、認証機能から承認結果(OK)を受け取る
4. システムは、サービスを実行する
5. システムは、認証機能に対し、「サービス利用状況を監視する」ユースケースを実行する
6. システムは、アクターに対し、サービスの利用を開始した旨を返す
7. UC を終了する

■例外フロー

1 .アクターは、システムに対し、サービスの利用を要求する
2. システムは、認証機能に対し、「サービスの実行を承認する」ユースケースを実行する
3. システムは、認証機能から承認結果(NG)を受け取る
4. システムは、アクターに対し、承認に失敗した旨を返す
5. UC を終了する

■課題や T．B．D 項目

なし

■備考

なし

＜UC03 名：サービスの実行を承認する＞

■概要

ユーザが指定のサービスを実行できる権限があるか確認する

■アクター

なし

■事前条件

ユーザが認証済みである

■事後条件

システムにサービス利用可否が渡されている

■メインフロー

1. システムは、認証機能に対し、指定ユーザが指定サービスを利用できるか確認する
2. 認証機能は、指定ユーザにサービス実行の権限があるかどうかを判断する
3. 認証機能は、システムに対し、サービス利用可否を渡す
4. UC を終了する

■課題や T．B．D 項目

なし

■備考

なし

＜UC04 名：サービス利用状況を監視する＞

■概要

ユーザがサービスを利用している状況を監視し、記録する

■アクター

なし

■事前条件

ユーザのサービス利用が承認されている

■事後条件

サービス利用状況の記録(課計)が存在している

■メインフロー

1.システムは、 認証機能に対し、サービス利用状況を通知する
2. 認証機能は、サービス利用状況を記録する
3. UC を終了する

■課題や T．B．D 項目

なし

■備考

なし

モデル一覧

着目点	コンセプト	ポイント
機能に着目したモデル	認証機能の設計として、広く知られている AAA モデルの 3 機能に着目したモデルです。	AAA モデルの 3 要素を基本クラスとしたモデルです。なんらかの指針を使った設計例として利用してください。
エンティティに着目したモデル	認証がユーザのサービス使用に対して制限を掛けるという観点から、制限を掛けるエンティティに着目したモデルです。	エンティティを中心に理解しやすいシンプルな作りにしたので、モデリング初心者の方にお勧めです。
状態に着目したモデル	ユーザが認証された・認証されていない、といった状態に着目したモデルです。	状態毎の振舞いの違いを、ステートパターンで実現します。
メタファを使ったモデル	なし	

AAA モデルとは

本章には、AAA モデルの用語が 3 つの設計モデルで共通概念として使用されています。

ここで、AAA モデルの概要を説明します。

AAA モデルとは、認証機能を実現する上で、主だった機能をつぎの 3 要素に分割したモデルを指します。

ユーザを認証する、適切なサービスを提供する、サービス使用の記録を作成する、これらを認証（Authentication）、承認（Authorization）、アカウンティング（Accounting）と分類し、3 つの頭文字 A をつなげて AAA モデルと呼ばれます。

また AAA モデルは、AAA プロトコルとも言われ、認証、承認、アカウンティングの 3 要素自体と、またその要素間の通信とにおいて秘匿性が設計上考慮されていることを指します。

本章に於いて、AAA のアカウンティングの呼称を、監視と改名しています。これはアカウンティングがユーザ情報（アカウント）と誤解しやすいためです。

機能に着目したモデル

モデリングのコンセプト

本モデルは、広くしられている AAA モデルの構造を模す事を出発点としました。AAA に登場する認証、承認、アカウンティングの３要素が取り扱う情報の責務に着目することを本モデルのコンセプトとして設計を行いました。

３要素のほかに要求仕様から読み取れる、ユーザの ID カード、ユーザに提供するサービス、ユーザの使用履歴といった情報を抽出し、それを元に分析作業を行いました。

分析モデル

静的モデル

クラス構成その１

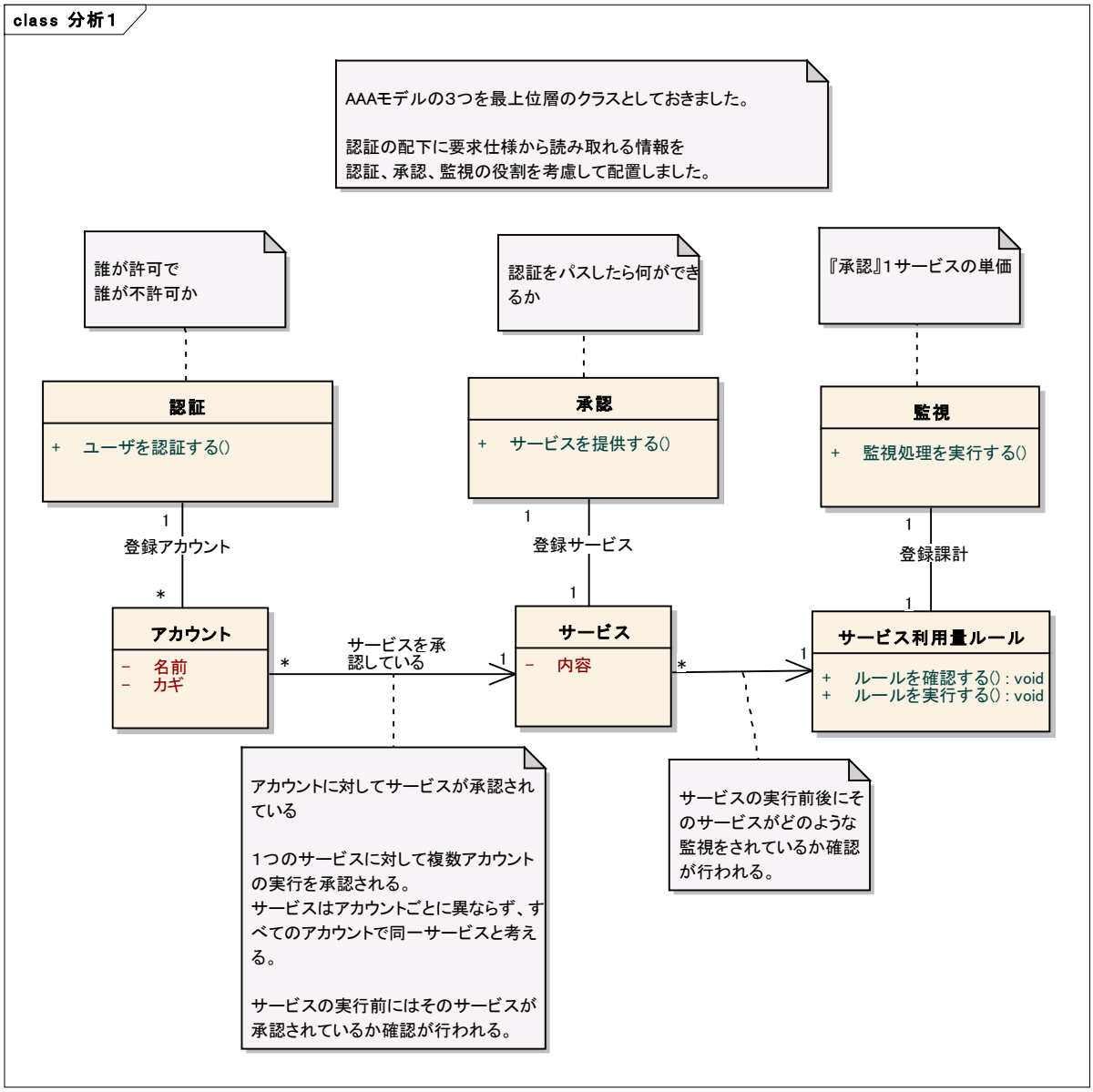


図2

最上段には AAA の３要素、認証、承認、監視を配置しました。

認証が扱う情報は、ユーザを識別するための情報です。認証が扱うユーザを識別する情報をアカウントとし、その下に配置しました。

承認が扱う情報は、サービスです。承認が扱うサービスを、その下に配置しました。

監視が扱う情報は、サービス利用（サービス利用後情報）です。

ひとまず分析作業はここまでとして、この後シーケンス検討、PIM モデル作成を通して必要なクラスを抽出します。

クラス構成その１の具体例

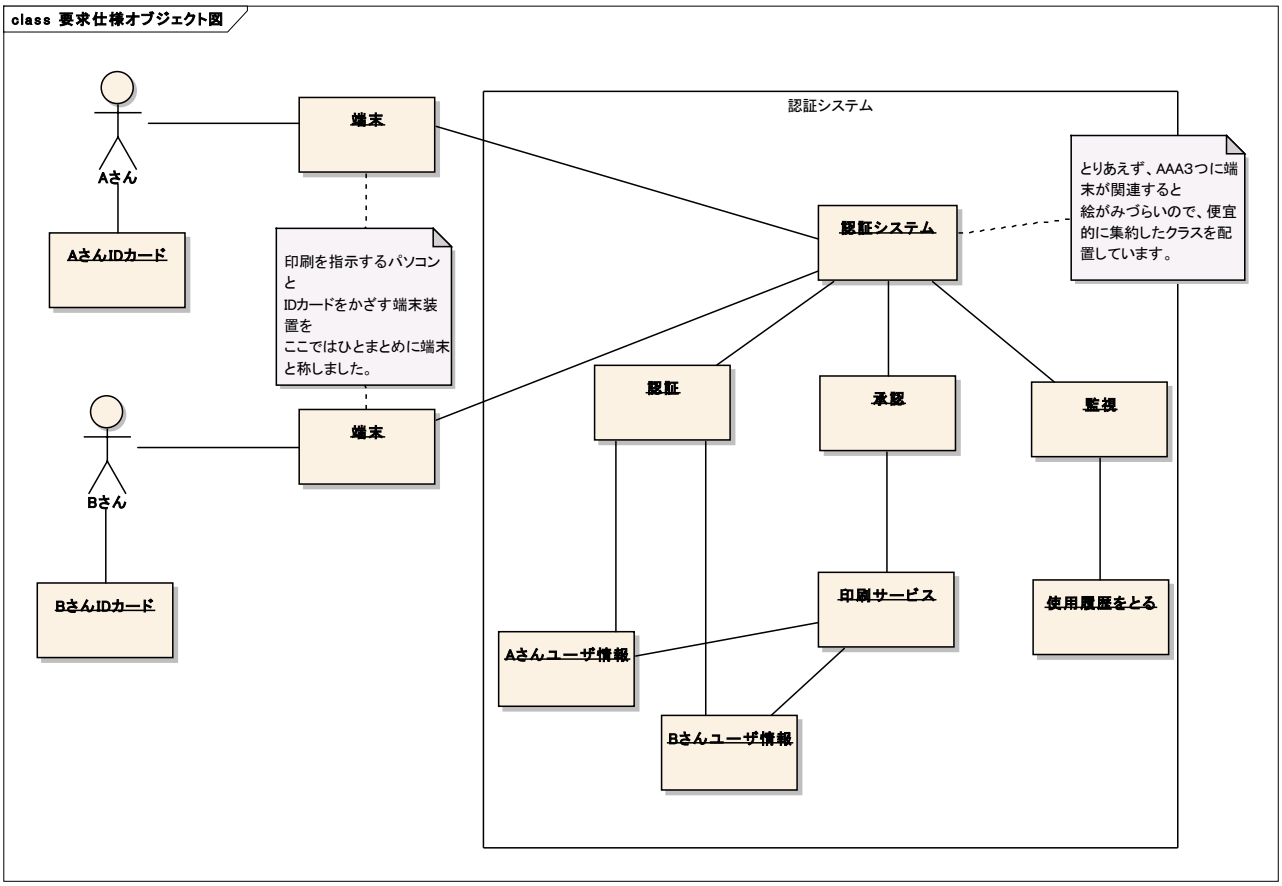


図3

要求仕様の状況をひとまずオブジェクト図にしてみました。クラス図その１の登場人物でひとまず足りている模様です。

クラス構成その２

クラス構成その１をもとに、さらにクラスを抽出

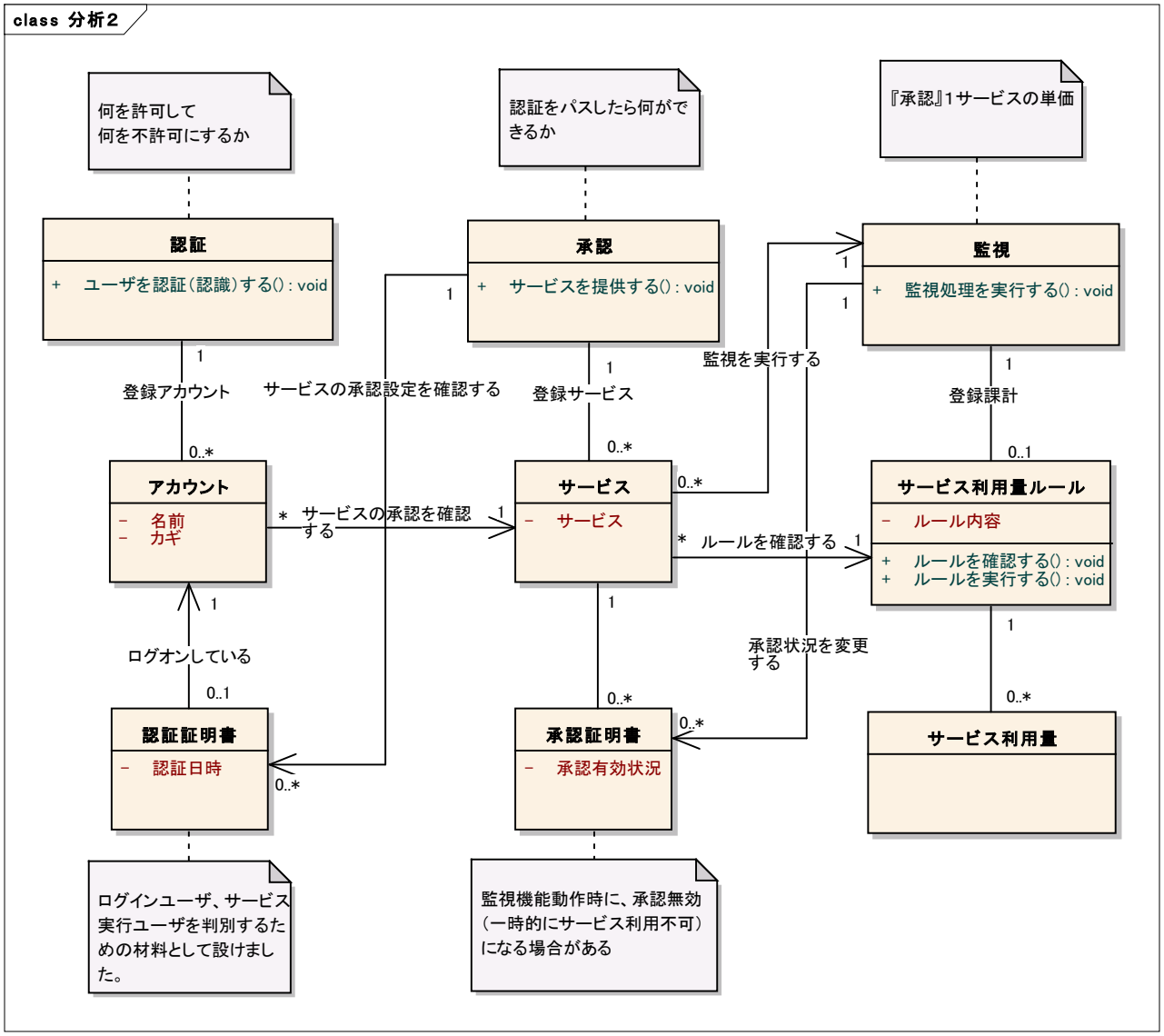


図4

シーケンスの検討や PIM モデルの検討を通してクラス図は上記のように変化しました。以下が追加された最下段のクラスの責務と説明となります。

認証証明書：ログインアカウントと非ログインアカウントを区別するためにログイン時に生成されます。

承認証明書：サービス実行前にサービスの実行権が監視からも認められているか確認するクラスです。例えば監視条件を満たさない場合（課金不足など）、サービスの実行権を失う状況を実現します。

サービス利用量：監視の使用記録などの実データとなります。

つぎにユースケースに対応したシーケンス図を示します。インタフェースは先の PIM モデルのものを利用しています。

動的モデル

ユーザを認証するときの相互作用

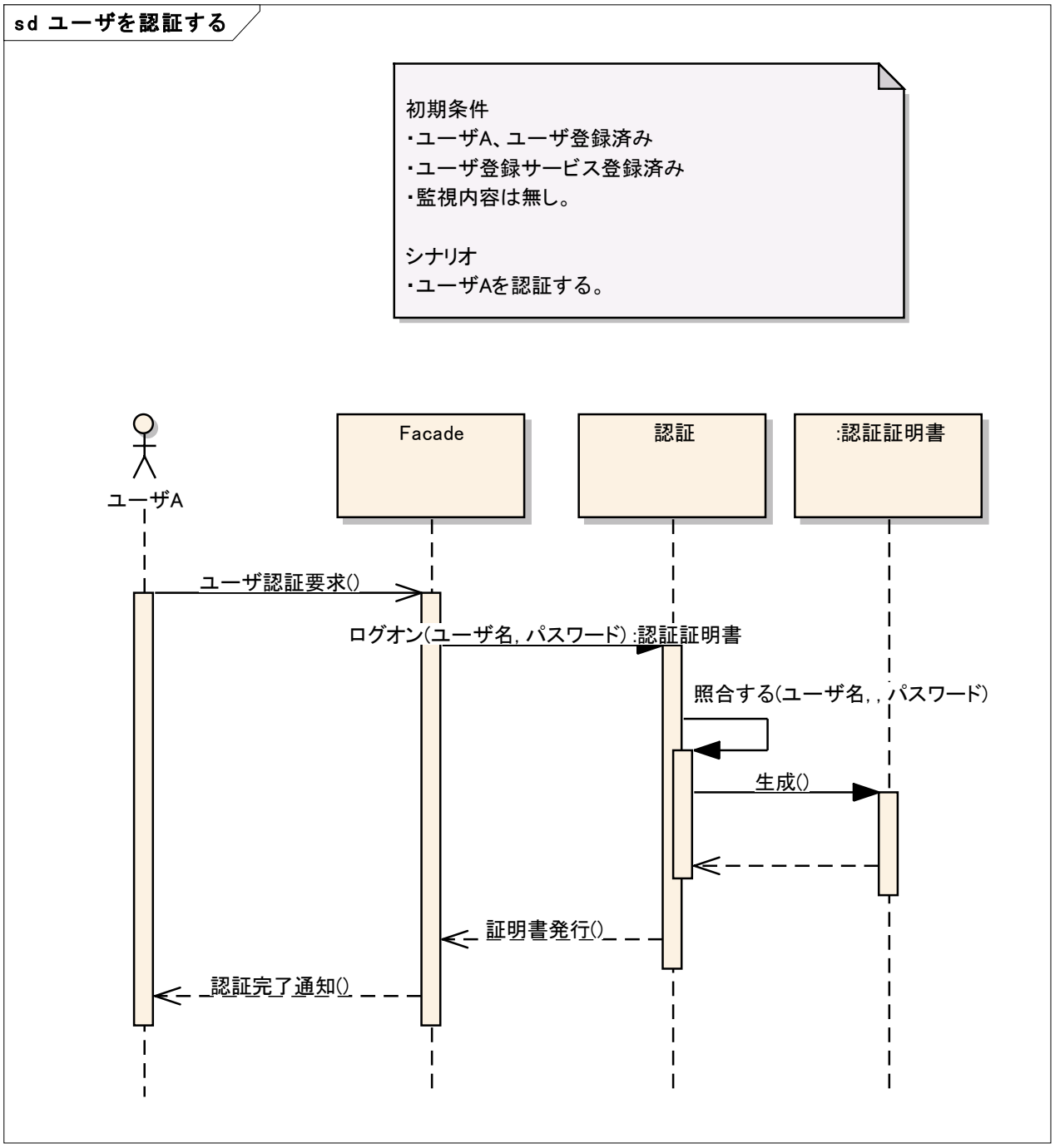


図5

サービスの実行を承認するときの相互作用

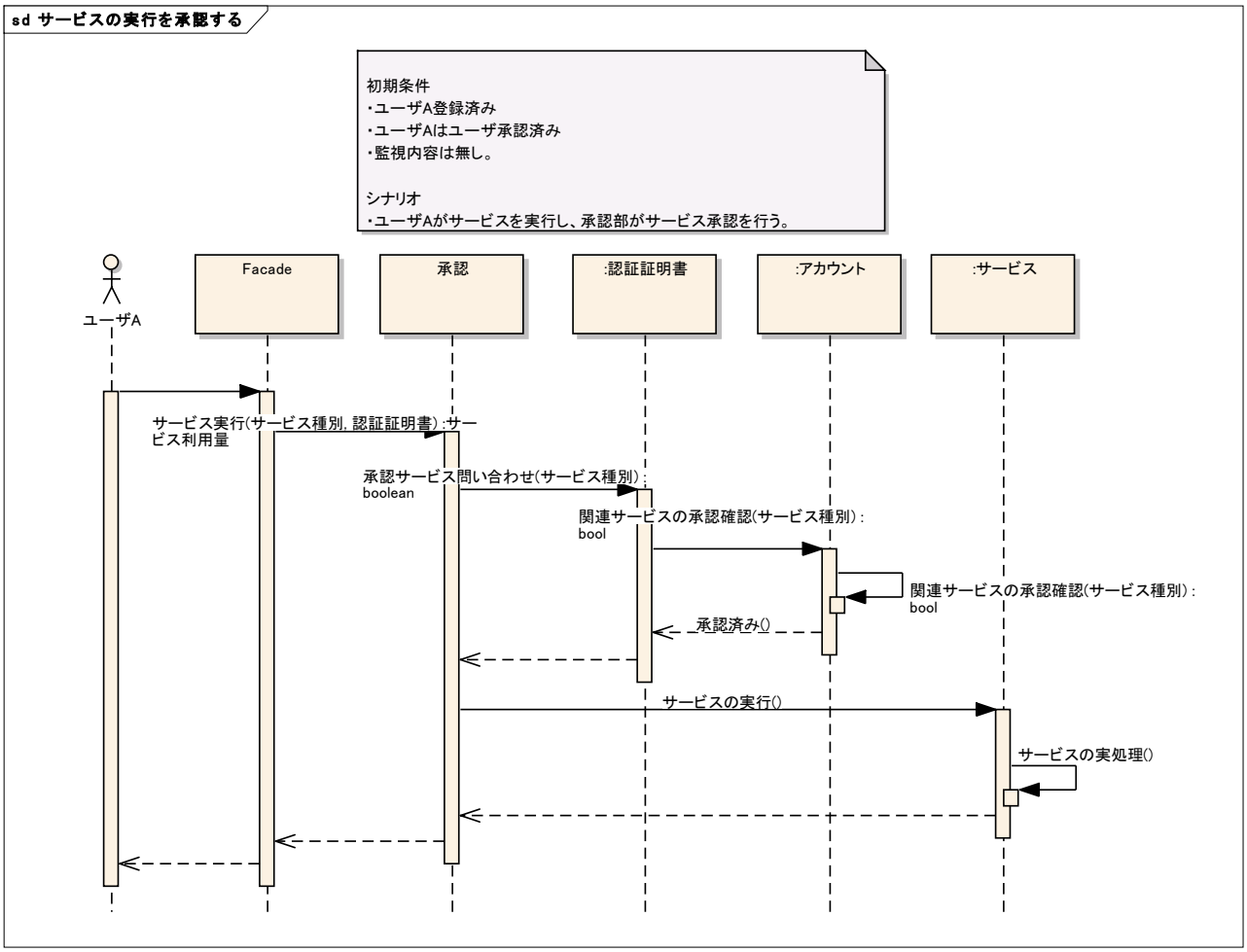


図6

サービスの利用状況を監視するときの相互作用

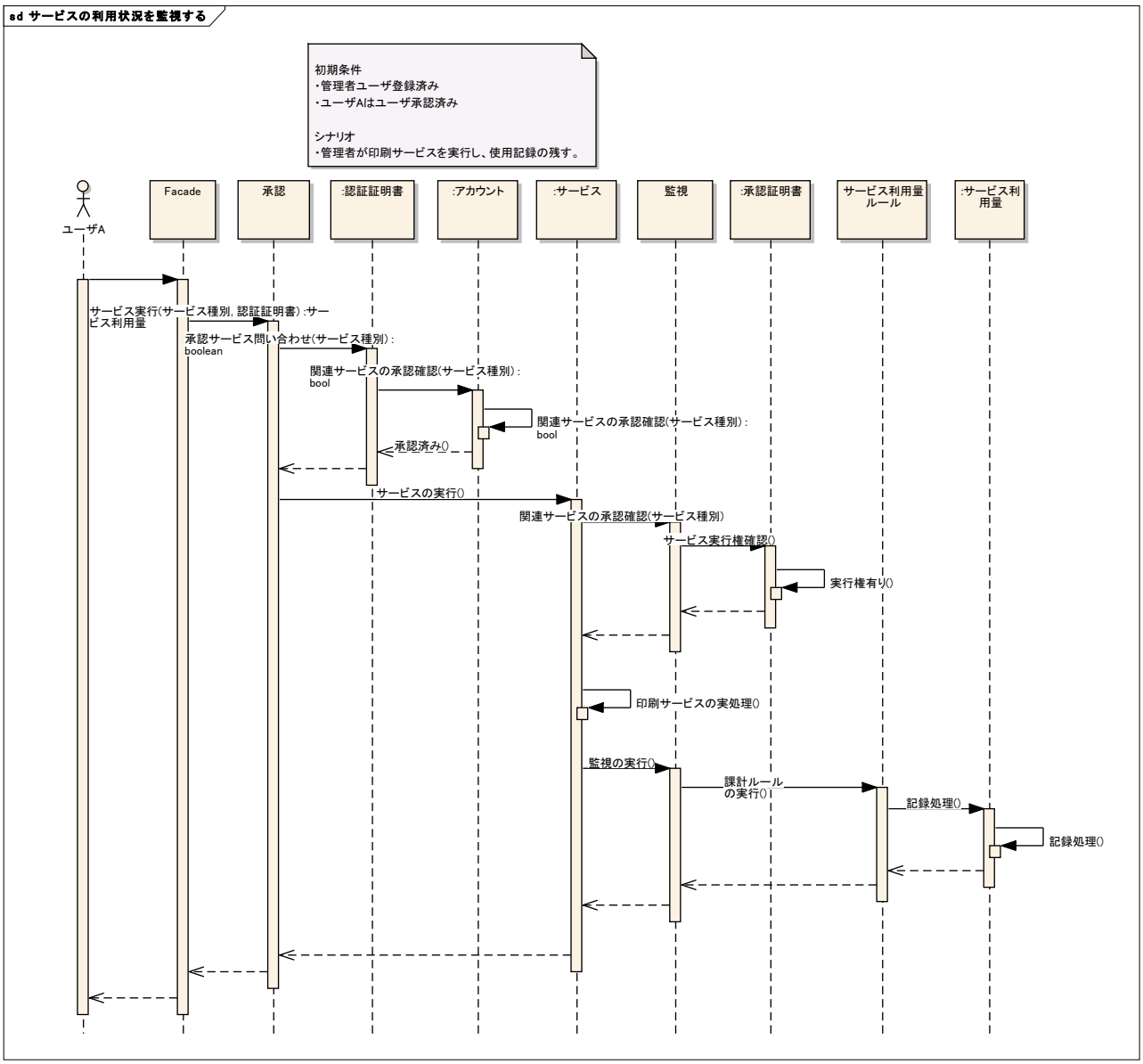


図7

PIM 設計モデル

静的モデル

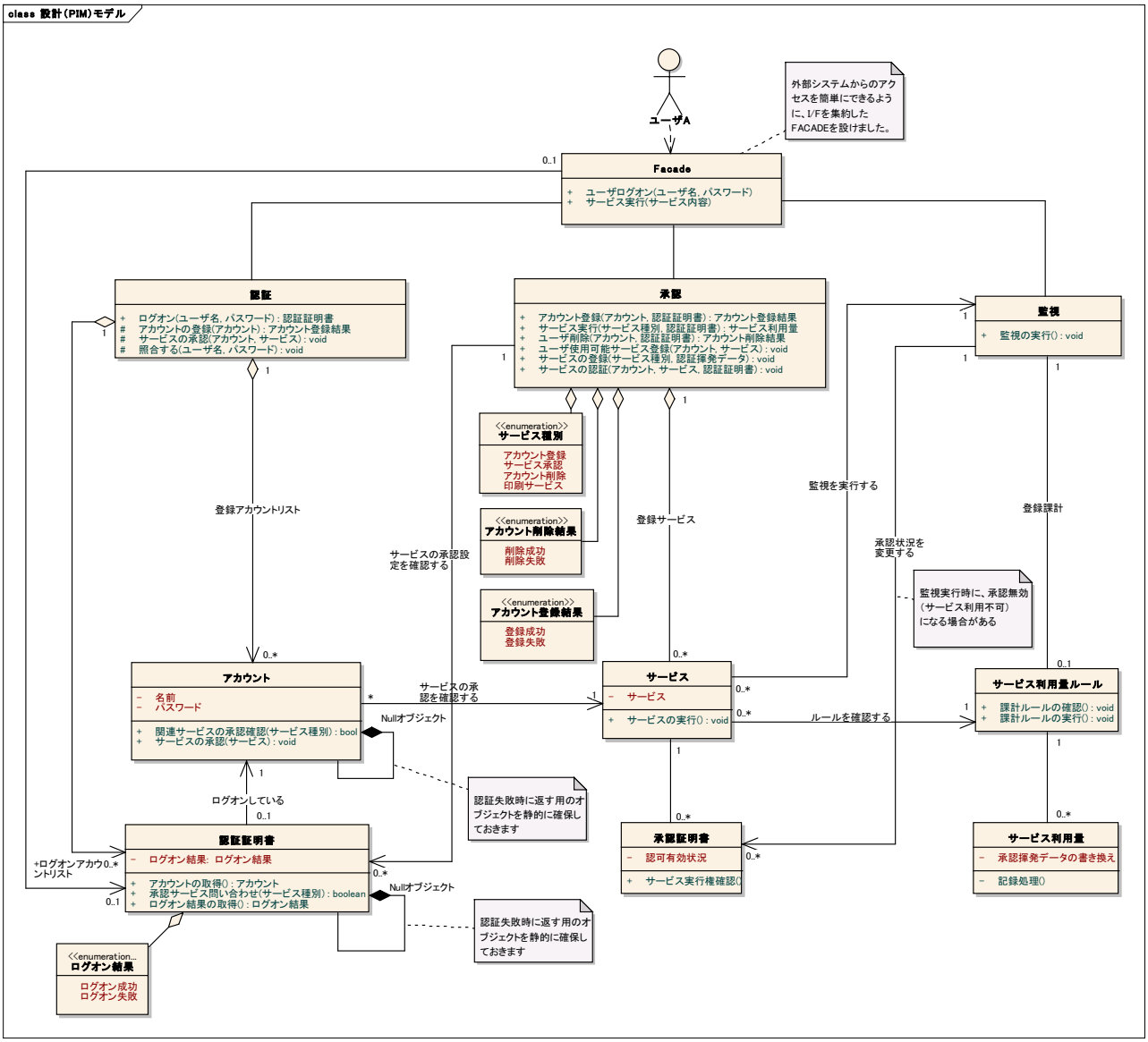


図8

エンティティに着目したモデル

モデリングのコンセプト

認証機能は、ユーザがサービスを使用したいという要求に対して、正当なユーザか？正当なサービスか？正しく使用されているか？といった観点からサービス使用に対して制限を掛けるという機能になります。この事からユーザがサービスを利用するまでに、ユーザにサービス利用の制限をかけるエンティティが幾つか出現すると考えられます。

これらのエンティティを明確化することをモデリングの出発点として分析作業を行いました。

分析モデル

静的モデル

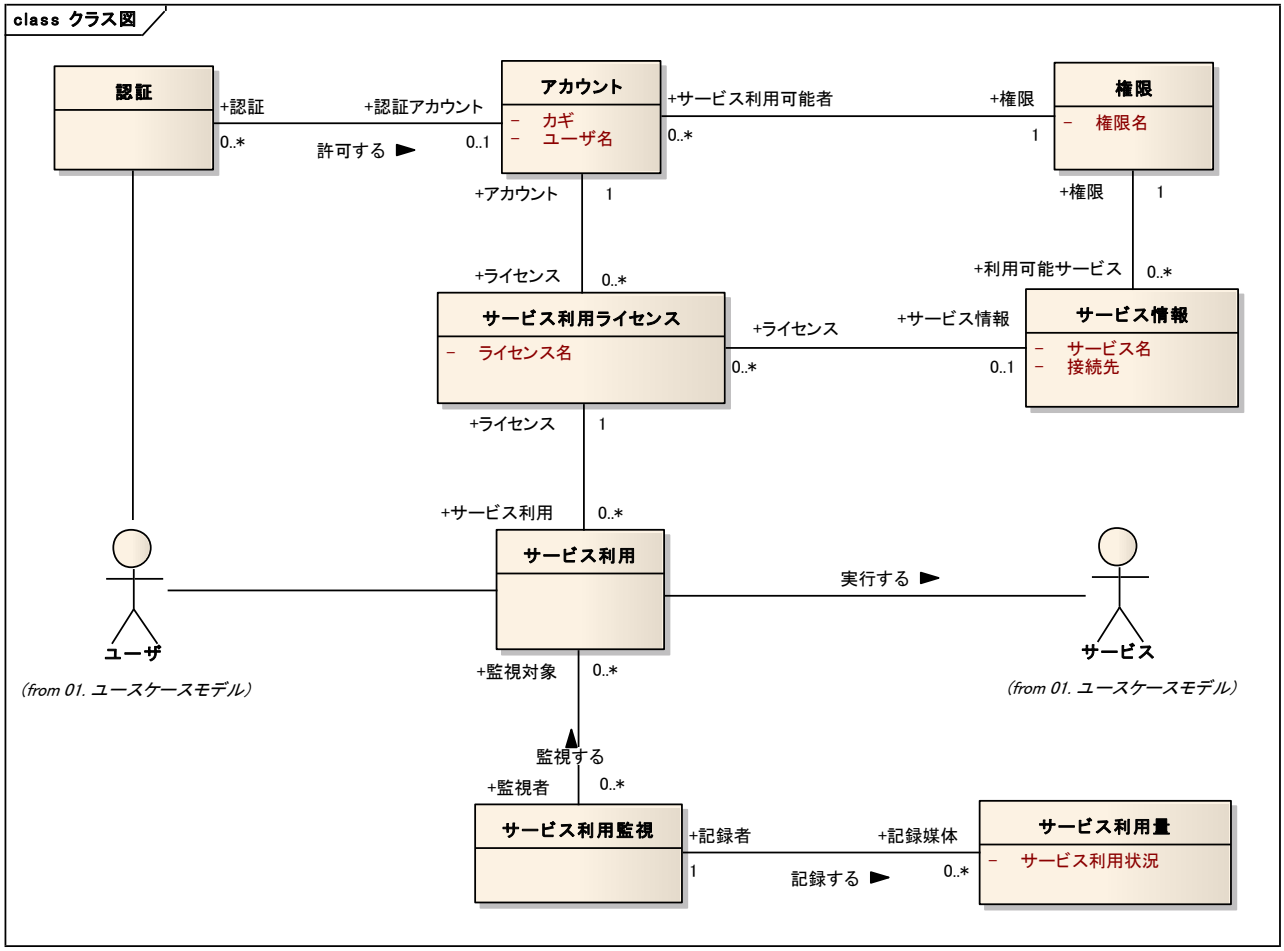


図9

このクラス図において、「認証」「サービス利用」「サービス利用監視」がそれぞれ認証、承認、監視の役割に当たります。これら以外の情報がサービス利用に制限をかけるためのエンティティとなります。制限をかけるためにどのような情報が必要かという点を検討するために、サービスの利用時に制限するために行う確認事項を洗い出してみます。

- 正当なユーザか？
- 正当なサービスか？
- 正当な組み合わせか？
- 正当に使用されているか？（監視記録）

このうち、正当な組み合わせに関しては、サービスの実行時刻や実行場所など様々な条件が考えられますが、モデルを簡素化するために、一旦、アカウントとサービスの組み合わせが適切か？というシンプルな確認とします。

上述した確認事項から残りのエンティティを見た場合、

・「アカウント」は、サービス利用者が妥当かどうかを判断するための情報

・「サービス情報」は、正当なサービスを実行するための情報

・「権限」は、どのユーザがどのサービスを実行できるかを示した情報

・「サービス利用量」は、サービスの利用状況の記録

という位置づけになります。これで先ほどの確認に伴うエンティティは網羅できました。

クラス図の中で残された一つの「サービス利用ライセンス」は、ユーザが特定のサービスの実行を許可された事を示す情報として作成しました。

サービス利用前のオブジェクト構成

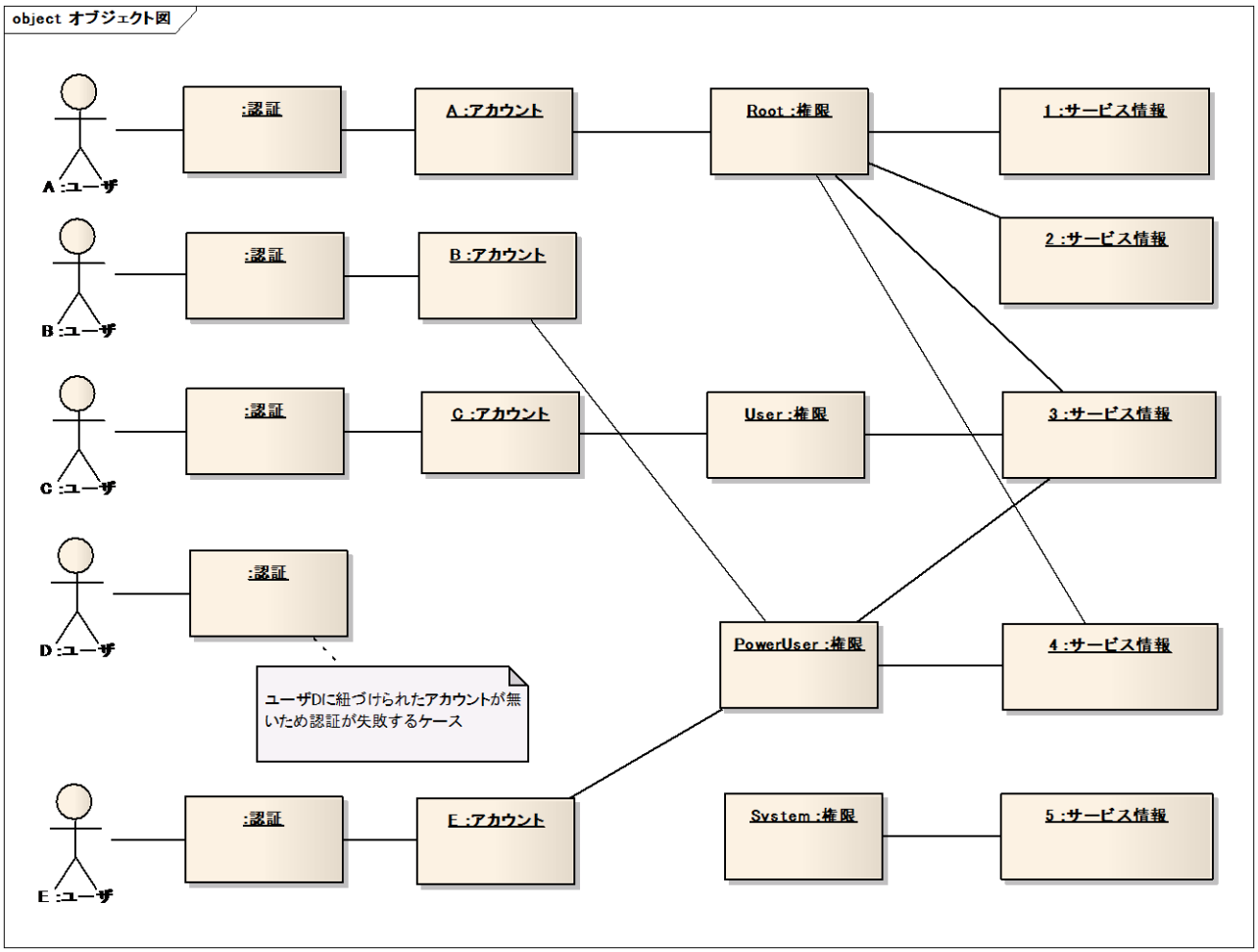


図10

サービス利用中のオブジェクト構成

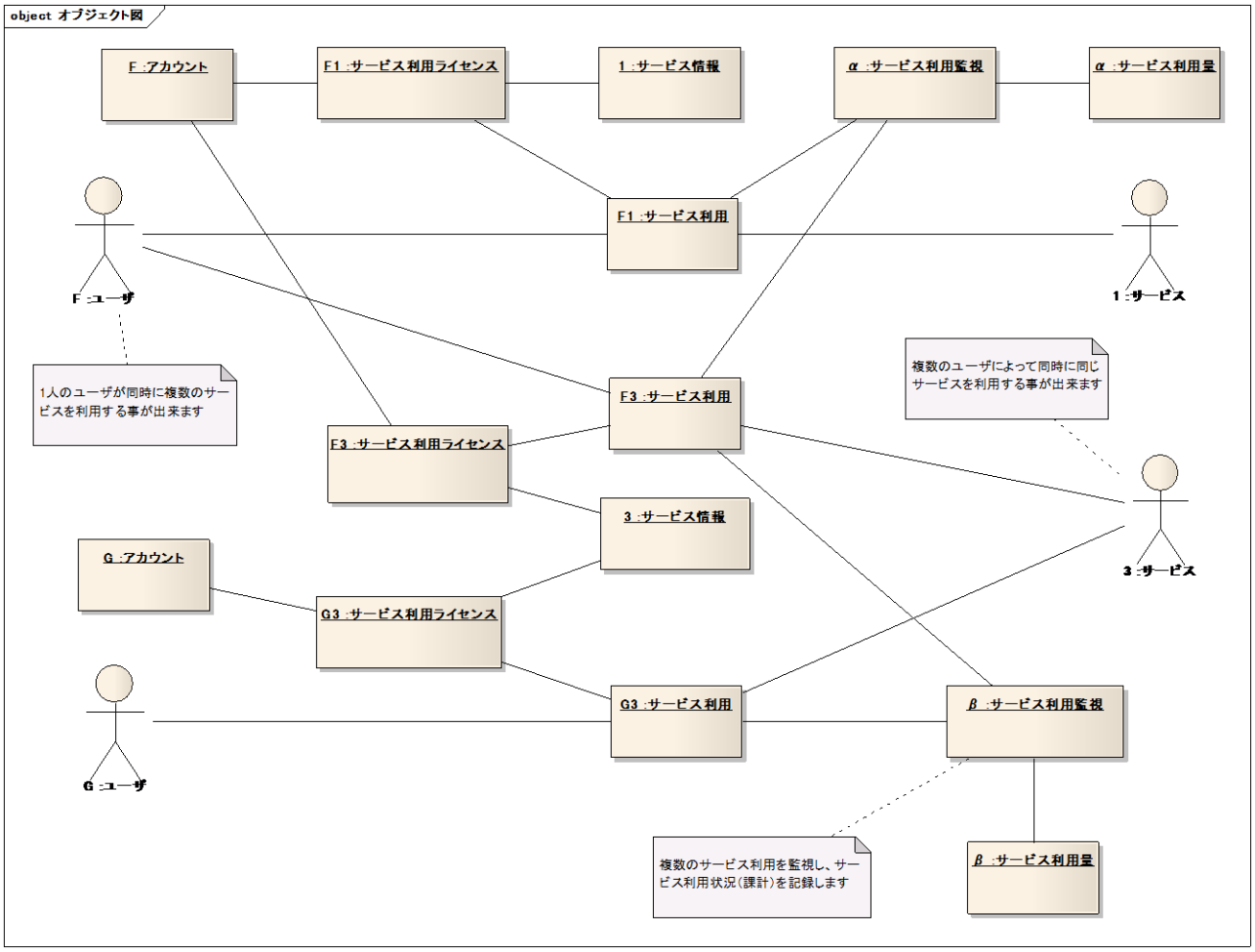


図11

最初のオブジェクト図が、ユーザから利用可能なサービスを特定するまでのオブジェクト構成を表したもので、二つ目のオブジェクト図が、利用可能なサービスが特定された後、サービスを利用して終了するまでのオブジェクト構成を表したものとなります。

「アカウント」「権限」「サービス情報」という関連は、認証が始まる前から既に存在しています。「認証」や「サービス利用」は、これらのエンティティや関連から、認証、承認といった制限を実現します。

「サービス利用ライセンス」と「アカウント」「サービス情報」間の関連は、承認が出来た後に作られる関連となります。また、「サービス利用監視」「サービス利用」間は、「サービス利用監視」の責務に応じて、様々な「サービス利用」の監視を行う事ができるような関連になっています。

動的モデル

ユースケースシナリオに基づいて作成したシーケンスは次の４つとなります。認証、承認、監視の３つの機能が、前述したエンティティで実現できていることが分かります。

ユーザを認証するときの相互作用

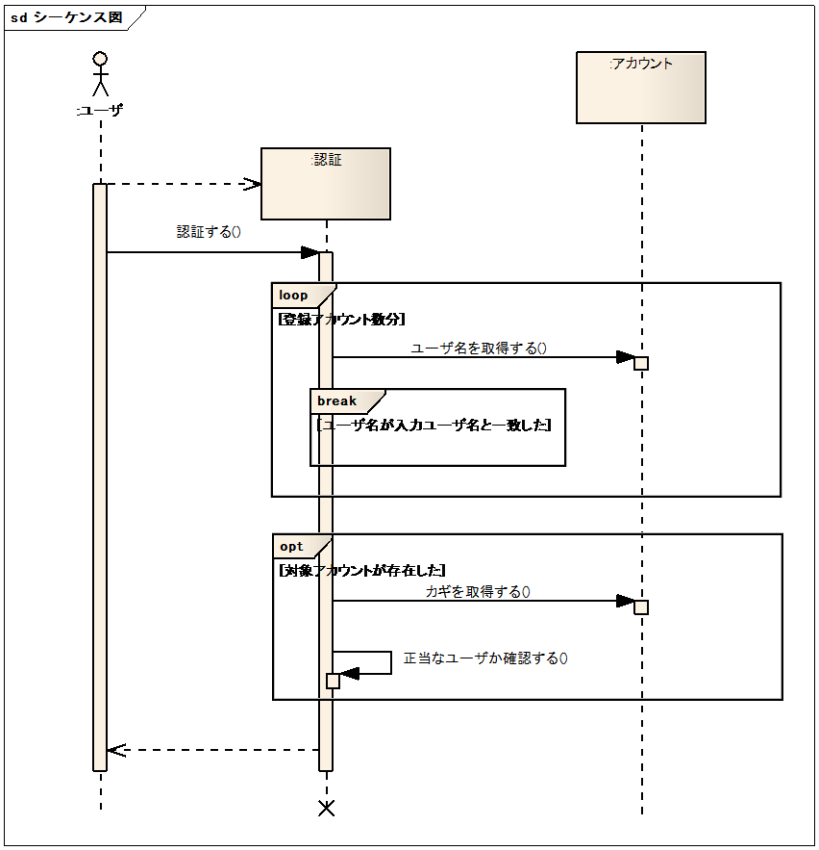


図12

サービスを利用するときの相互作用

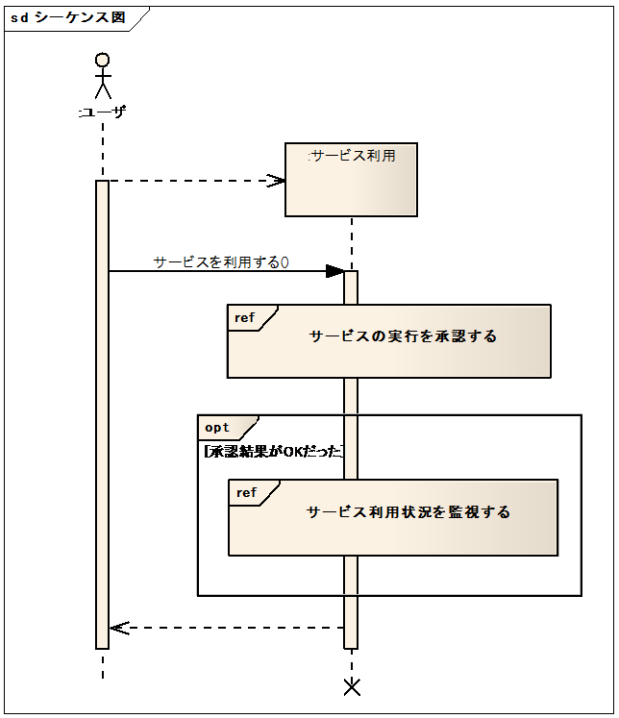


図13

サービスの実行を承認するときの相互作用

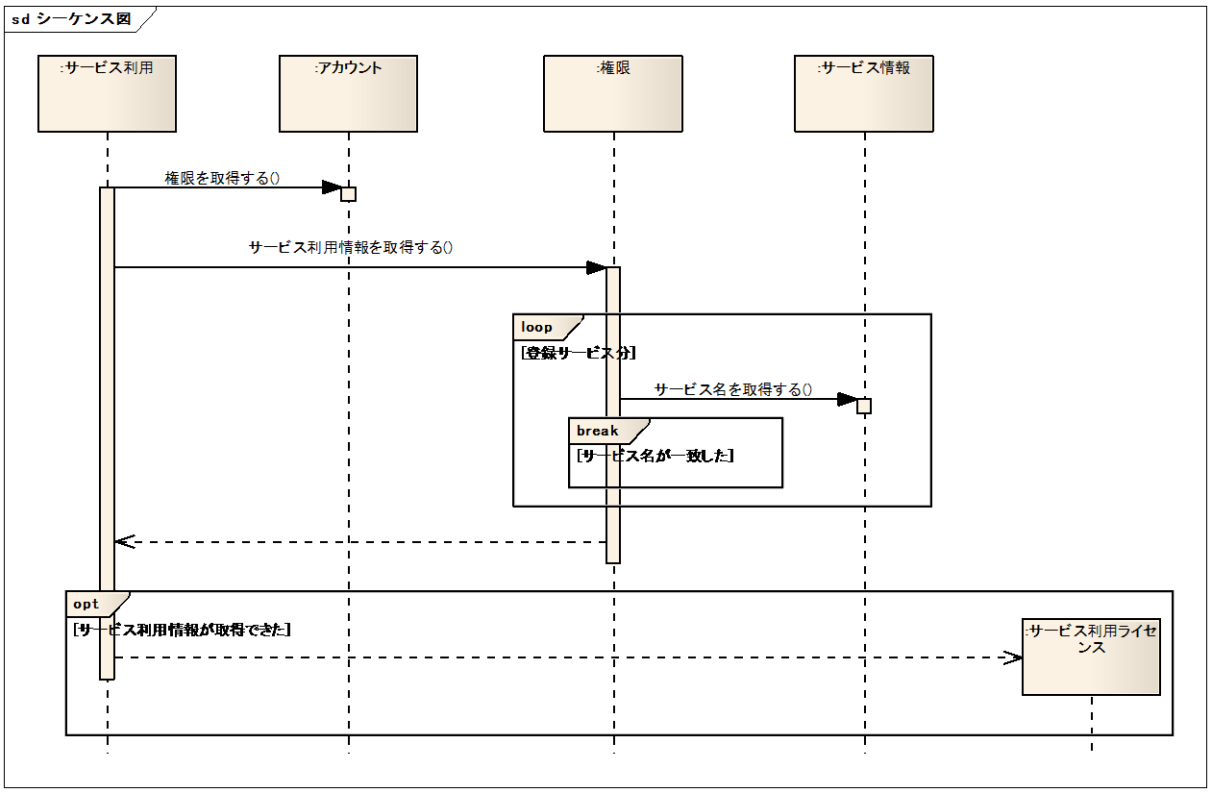


図14

サービス利用状況を監視するときの相互作用

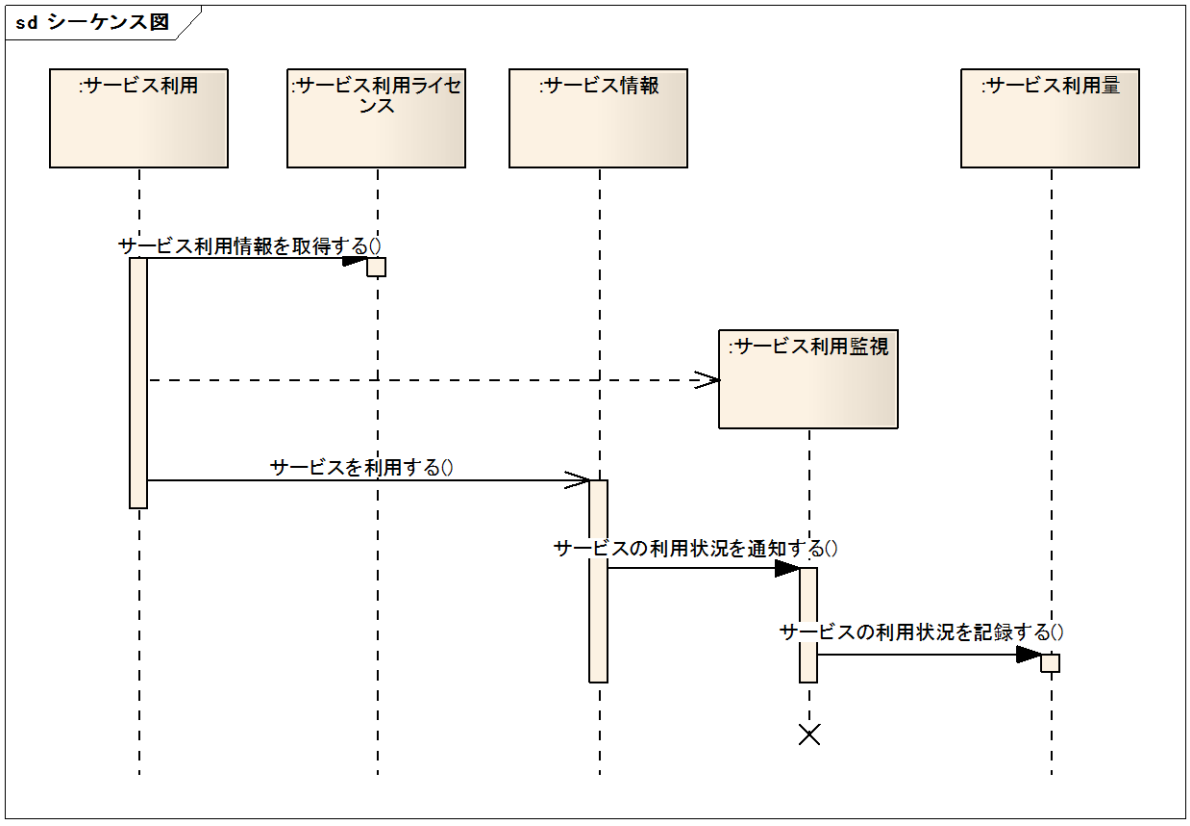


図15

PIM 設計モデル

分析モデルでは、ユーザがサービスを利用するまでに出現するエンティティを明確化することをモデリングの出発点としました。PIM 設計モデルでは、分析モデルで明確になったクラスに対して、責務をインタフェース（Boundary）、機能（Control）、データ（Entity）という 3 つの視点で分割することをモデリングの出発点とします。

このような構造にすることにより、例えば認証の場合、PC 画面からのパスワード認証であっても、指紋認証などのように機器デバイスからの認証であっても、ユーザインタフェース部分を変更するだけで機能部分に手を加えることなく対応する事ができます。

静的モデル

分析モデルで出たクラスを、インタフェース・機能・データに当てはめると次のようになります。

- 「認証」「サービス利用」は、インタフェース＋機能
- 「サービス利用監視」は、機能
- 「アカウント」「サービス情報」「権限」「課計」「サービス利用ライセンス」は、データ

このため、「認証」と「サービス利用」は、それぞれ次のように責務を分離しました。

- 「認証窓口」「サービス利用窓口」「サービスプロキシ」は、インタフェース
- 「認証」「サービス利用」は、機能

また、「アカウント」「サービス情報」「権限」の静的なデータは、どこか格納しておくものが必要になります。そのため、それぞれ一覧という名前をつけたクラスを導入しました。

さらに機能の結果保持用のクラスを導入して、操作を追加すると次のようなクラス図となります。

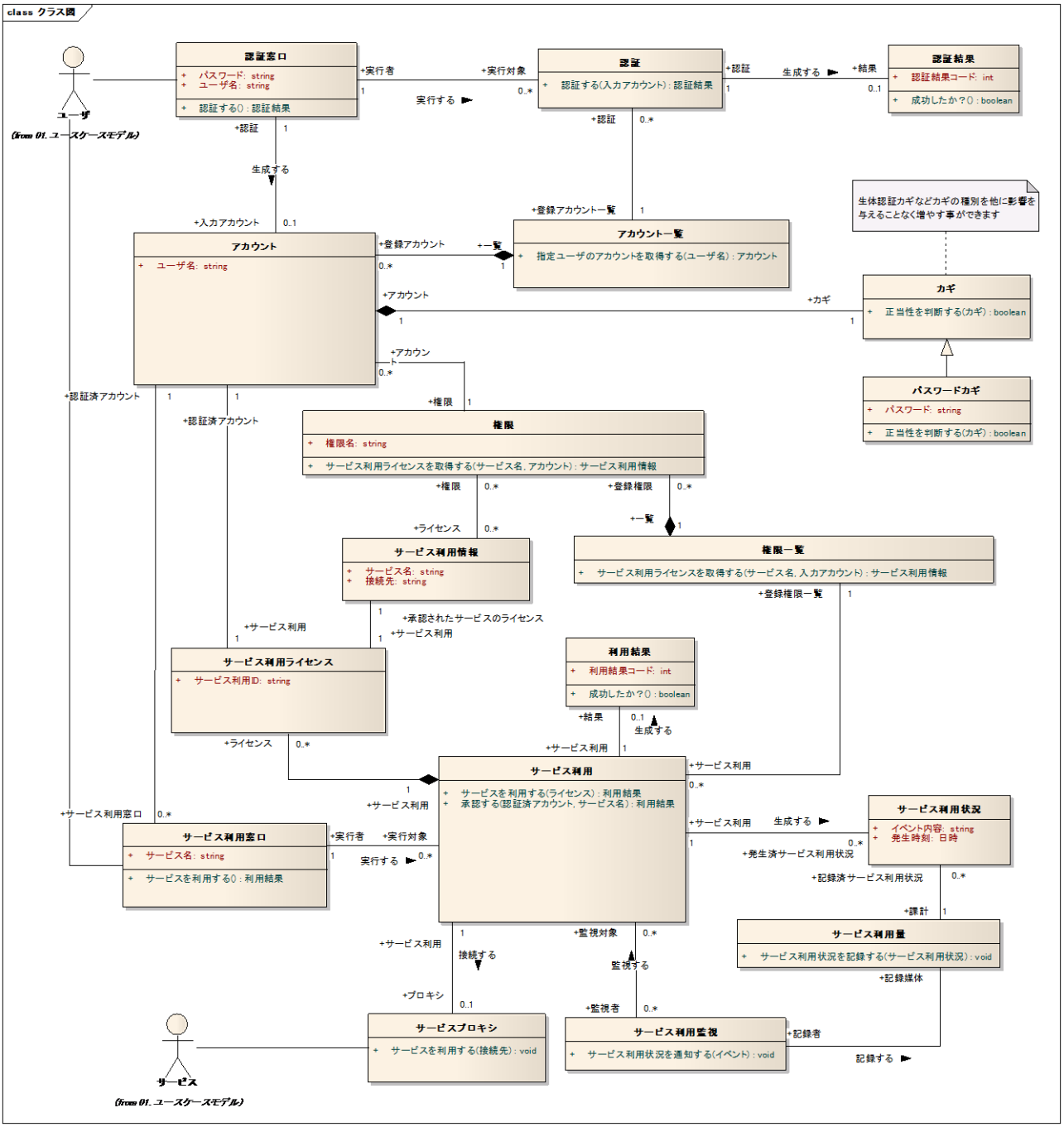

class クラス図
ユーザ
(from 01.ユースケースモデル)
認証窓口
+ パスワード: string
+ ユーザ名: string
+ 認証する() : 認証結果
+実行者
+実行対象
実行する
認証
+ 認証する(入力アカウント) : 認証結果
+認証
生成する
+結果
認証結果
+ 認証結果コード: int
+ 成功したか?() : boolean
+認証
+入力アカウント
生成する
アカウント
+ ユーザ名: string
+登録アカウント
+一覧
アカウント一覧
+ 指定ユーザのアカウントを取得する(ユーザ名) : アカウント
+登録アカウント一覧
生体認証カギなどカギの種別を他に影響を与えることなく増やす事ができます
カギ
+ 正当性を判断する(カギ) : boolean
パスワードカギ
+ パスワード: string
+ 正当性を判断する(カギ) : boolean
+アカウント
+カギ
+アカウント
+認証済アカウント
+認証済アカウント
+権限
権限
+ 権限名: string
+ サービス利用ライセンスを取得する(サービス名, アカウント) : サービス利用情報
+権限
+ライセンス
+登録権限
+一覧
サービス利用情報
+ サービス名: string
+ 接続先: string
権限一覧
+ サービス利用ライセンスを取得する(サービス名, 入力アカウント) : サービス利用情報
+登録権限一覧
+承認されたサービスのライセンス
+サービス利用
+サービス利用
+サービス利用
サービス利用ライセンス
+ サービス利用ID: string
利用結果
+ 利用結果コード: int
+ 成功したか?() : boolean
+結果
生成する
+ライセンス
+サービス利用
サービス利用
+ サービスを利用する(ライセンス) : 利用結果
+ 承認する(認証済アカウント, サービス名) : 利用結果
+サービス利用
+サービス利用
+サービス利用窓口
サービス利用窓口
+ サービス名: string
+ サービスを利用する() : 利用結果
+実行者
+実行対象
実行する
+サービス利用
生成する
サービス利用状況
+ イベント内容: string
+ 発生時刻: 日時
+発生済サービス利用状況
+記録済サービス利用状況
+課計
サービス利用量
+ サービス利用状況を記録する(サービス利用状況) : void
+記録媒体
+サービス利用
接続する
+監視対象
監視する
+プロキシ
+監視者
サービス
(from 01.ユースケースモデル)
サービスプロキシ
+ サービスを利用する(接続先) : void
サービス利用監視
+ サービス利用状況を通知する(イベント) : void
+記録者
記録する

図16

動的モデル

分析モデルをベースに PIM 設計モデルのクラス構造に基づいて作成したシーケンス図が次の４つの図になります。アクターがユーザインタフェースの責務を持つクラスとしかアクセスしていません。

ユーザを認証するときの相互作用

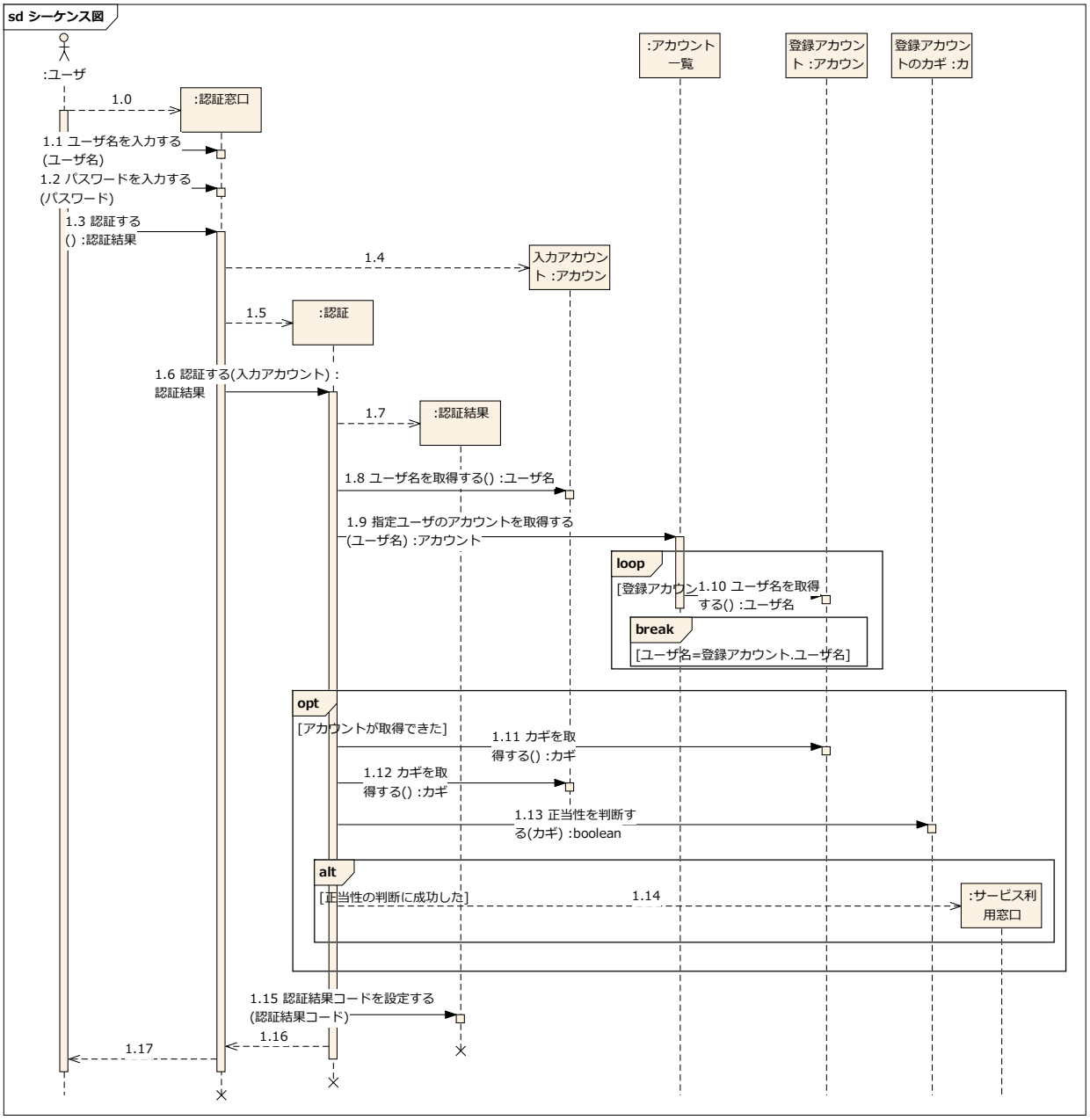


図17

サービスを利用するときの相互作用

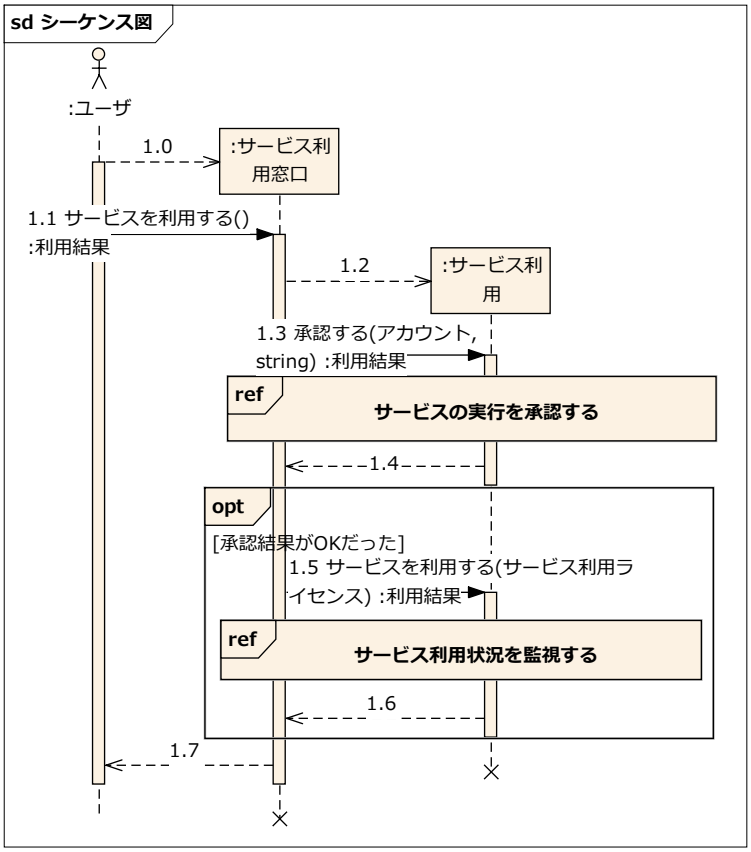


図18

サービスの実行を承認するときの相互作用

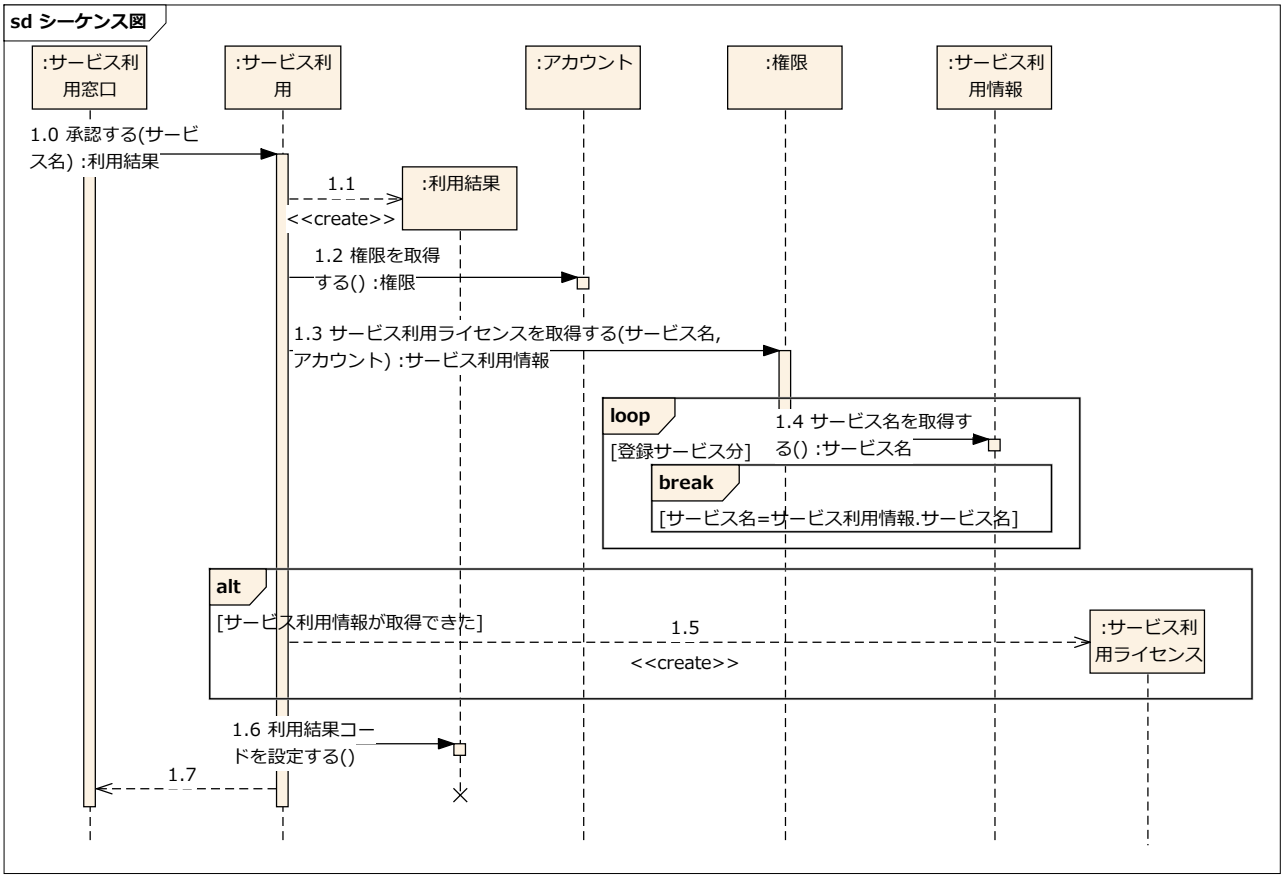


図19

サービス利用状況を監視するときの相互作用

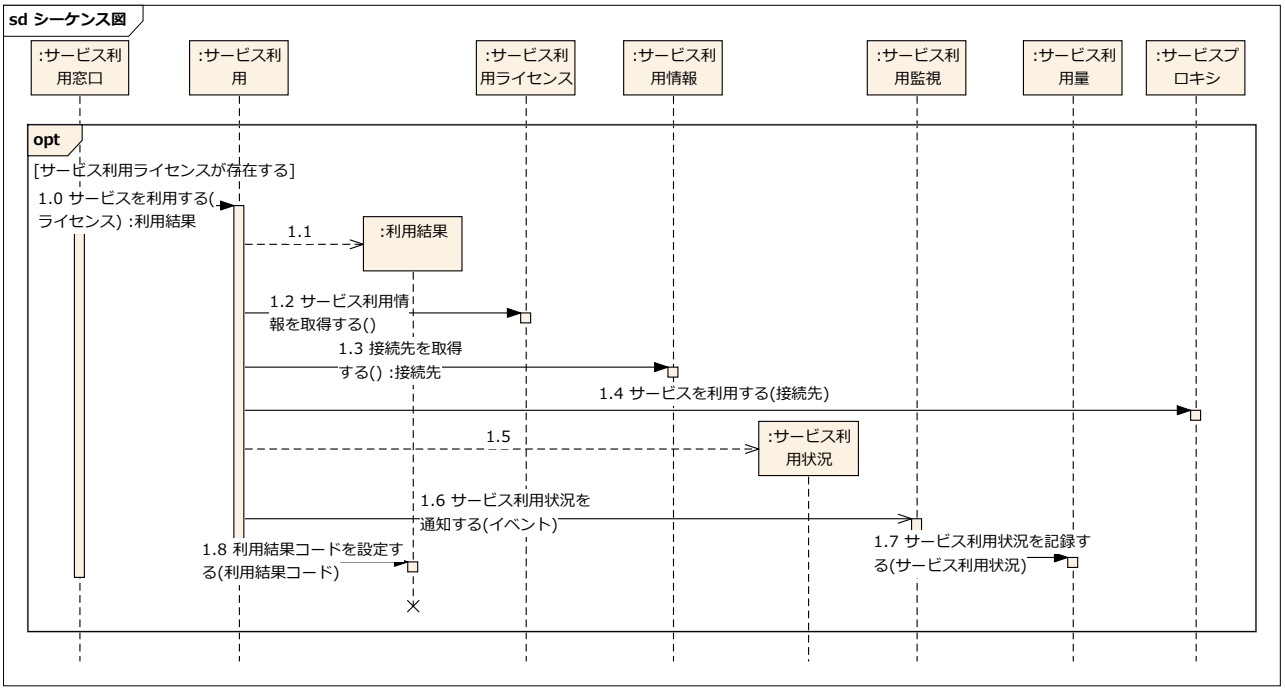


図20

状態に着目したモデル

モデリングのコンセプト

ユーザが認証された・認証されていない、といった状態に着目したモデルです。

認証モデルは、ユーザが認証されているか・認証されていないか、によってユーザからの処理依頼に対する振舞いを適切に切替えることが重要です。振舞いの切替えに、ヌケ・モレが無いことを、コーディング時ではなくて、構造設計レベルで抑えようというのがこのモデルのコンセプトです。

分析モデル

静的モデル

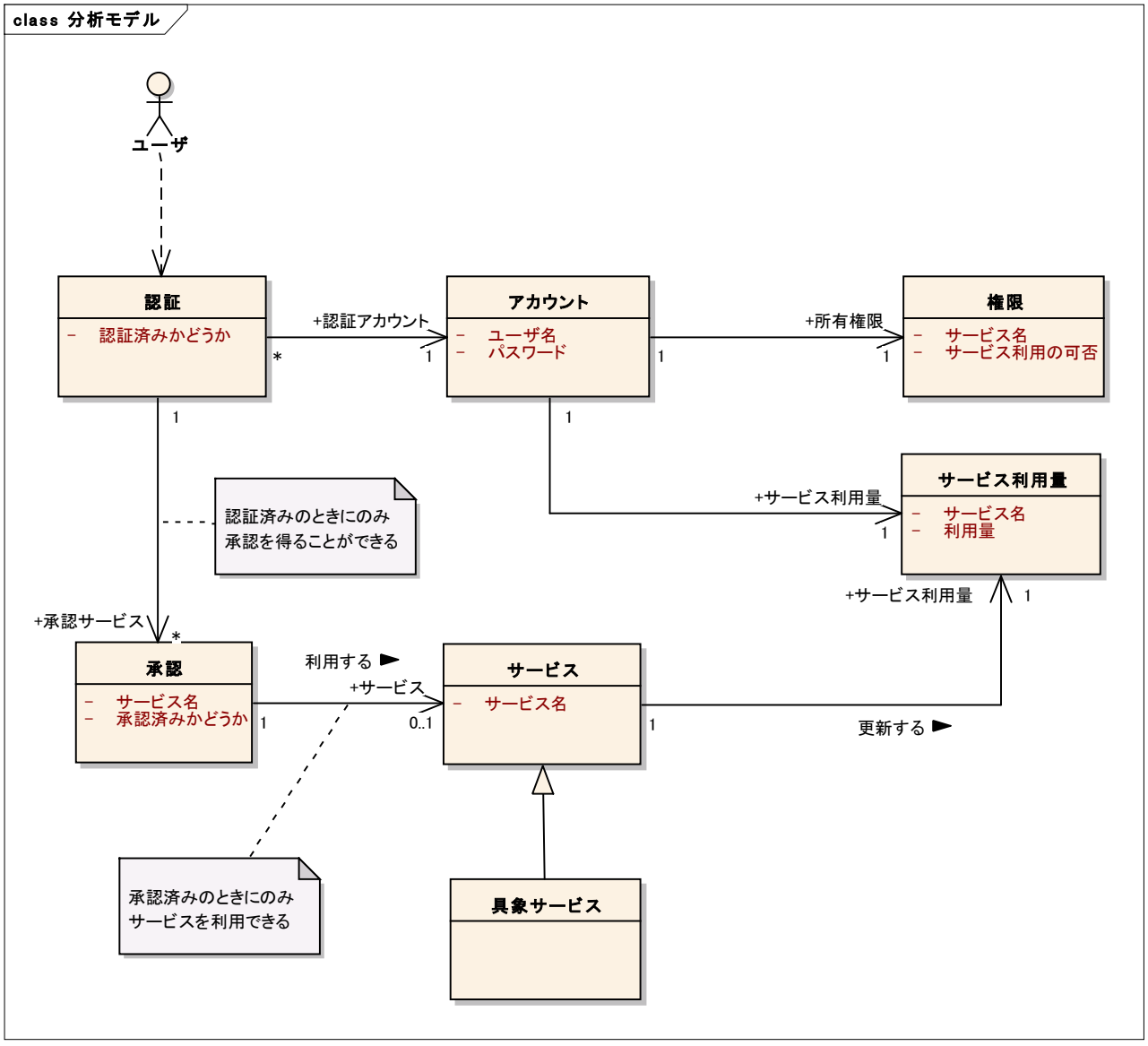


図21

ユーザとの認証に絡む振舞いを担うクラスとして「認証」を設けます。

ユーザ毎の情報を保持するクラスとして「アカウント」を設けます。「アカウント」には、認証に必要な情報「権限」と、そのアカウントが過去にサービスを利用した記録である「サービス利用量」を関連付けます。

ユーザが認証されたときに利用出来るサービスは複数あるため、サービス毎に承認が必要と考え、サービス毎の承認ずみかどうかを表すために「承認」を設けます。

ユーザのサービスを利用によって使用量を監視・記録するために「サービス」を設けます。各々のサービス固有処理は、「サービス」から派生した「具象サービス」が担うものとします。

ユーザによるサービス利用を監視した結果をユーザ毎に分けて記録するために、「サービス」からも、「アカウント」に関連づけた「サービス利用量」に関連を持たせます。

動的モデル

状態遷移

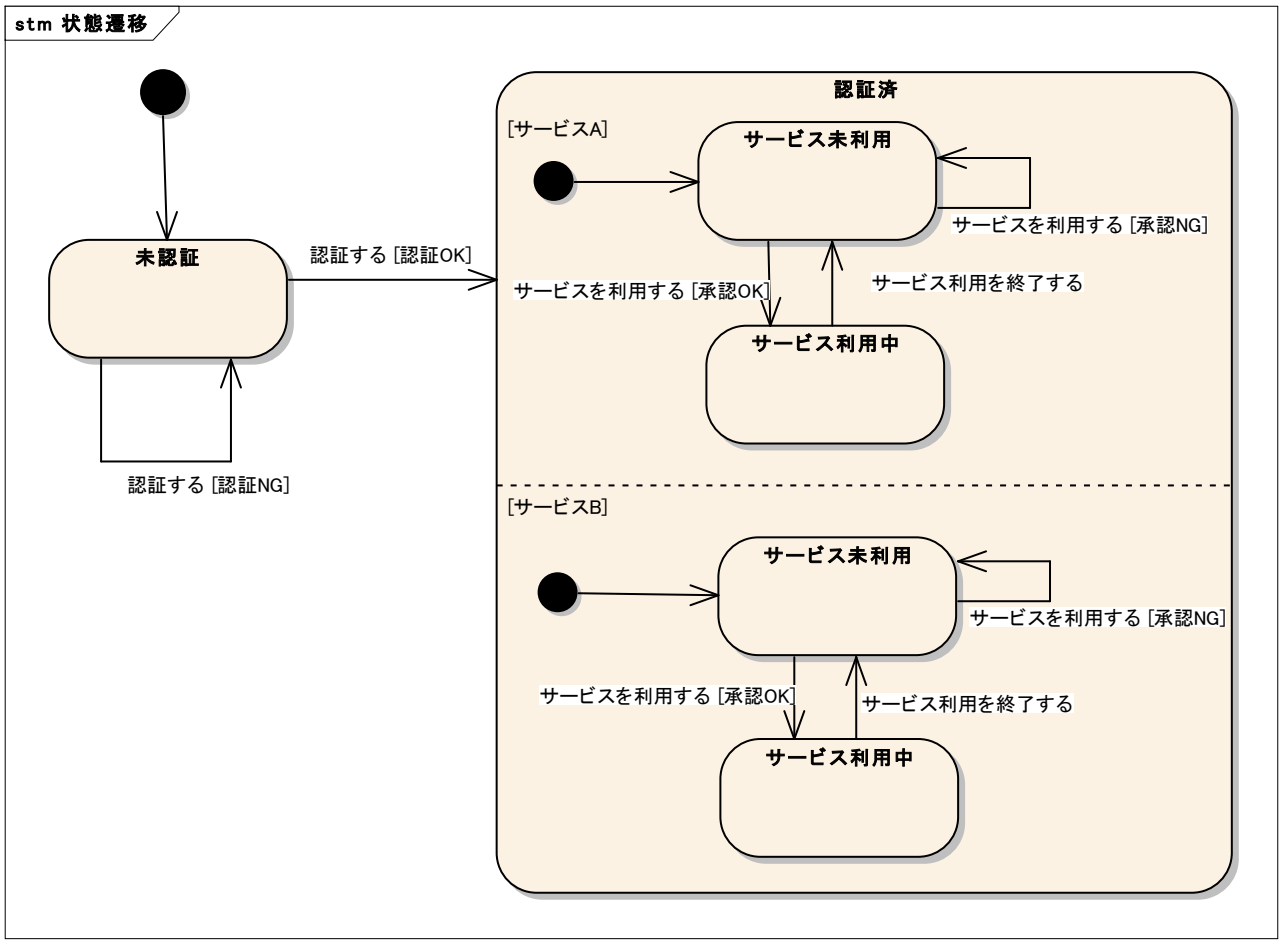


図22

認証モデル全体が、ユーザ認証、サービス利用承認によりどのように状態が変化するかを示します。

ユーザが認証されていない状態を「未認証」、ユーザが認証された状態を「認証済」と呼ぶことにます。最初の状態は「未認証」です。ユーザが認証を試みて、システムに正当なユーザであると判断されると「認証済」へ遷移します。

「認証済」状態にネストする形で、サービスを利用していない「サービス未利用」と、サービスを利用中の「サービス利用中」の 2 つの状態があります。「認証済」に遷移した直後は、「サービス未利用」です。これらの状態は、サービス毎にあります。ステートマシン図では、サービス A とサービス B の 2 つのサービスについて、それぞれ状態を持てることを示しています。

ユーザがサービスを利用しようとすると、システムが認可するかどうか判断します。システムが承認した場合は「サービス利用中」に遷移します。システムが承認しない場合は、「サービス未利用」に留まります。

ただし、「サービス未利用」と「サービス利用中」の状態は、システムに 1 つではなく、サービス毎に状態をもつ必要があります。

以下に、各状態における振る舞いの違いを示します。振る舞いが未定義の部分は―とし、「何もしない」とします。

状態		イベント		
		ユーザを認証する	サービスを利用する	サービスの利用終了
未認証		認証できれば「認証済」へ遷移	―	―
認証済	サービス未利用	―	承認できれば、サービスを起動し、「サービス利用中」へ遷移	―
	サービス利用中	―	―	サービスを終了し、「サービス未利用」へ遷移

例えば、「ユーザを認証する」は、「未認証」のときしか動作せず、それ以外の「認証済」状態では何もしません。

PIM 設計モデル

未認証・認証済、サービス未利用・サービス利用中、それぞれの状態を表すために、デザインパターンの 1 つ、ステートパターンを適用します。

ステートパターンは、振舞いに関するデザインパターンの 1 つです。ステートパターンを用いることで、状態によって振る舞いを変えるための条件分岐が不要になります。また、状態遷移図に対して状態遷移表がそうであるように、状態毎の振舞いの違いを考えるときのモレ・ヌケを防止する効果があります。

静的モデル

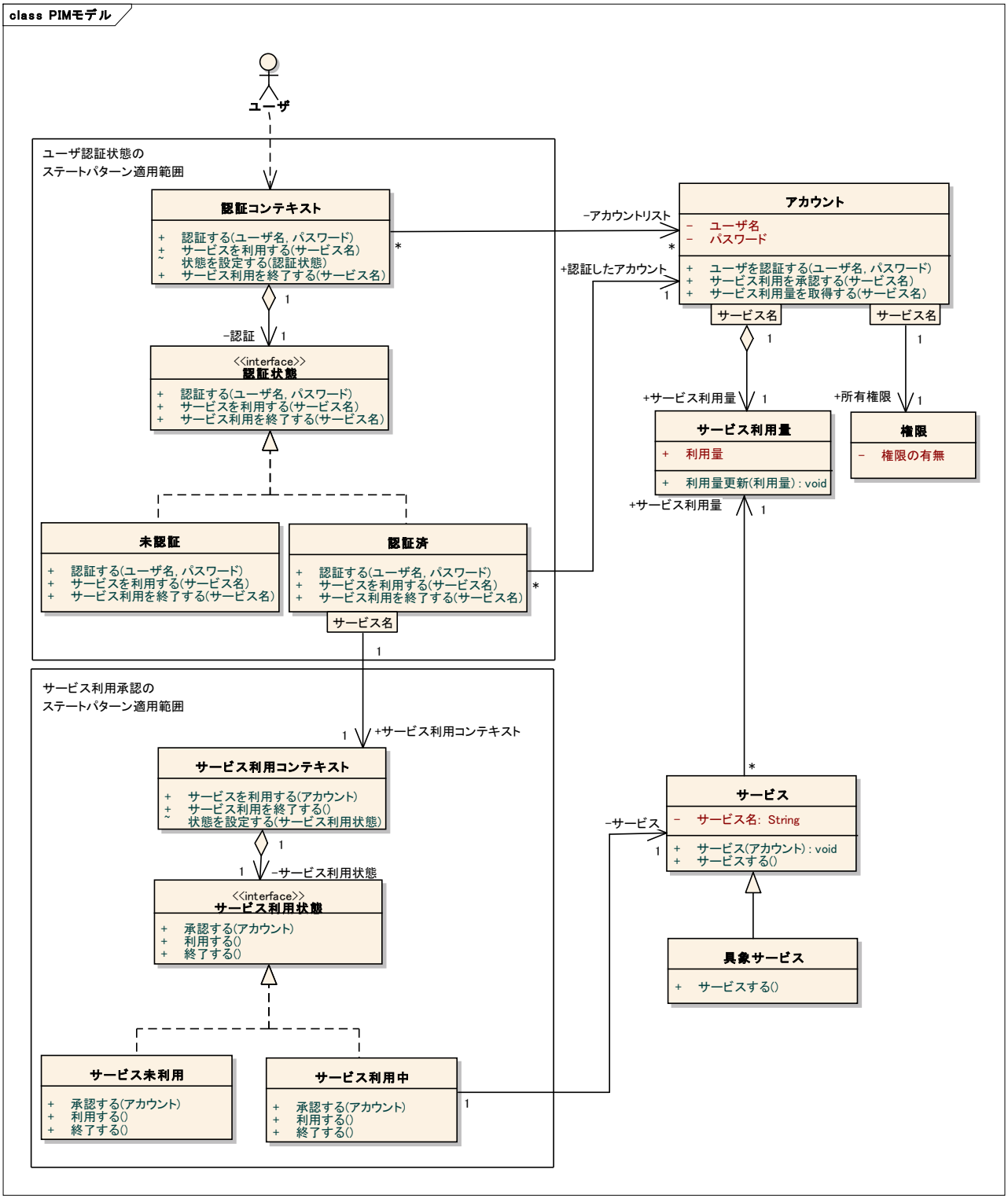


図23

システムに登録されたユーザ情報である「アカウント」には、サービス毎にそのサービスを利用可能か示す「権限」と、サービスを利用したときに計上される「サービス利用量」を関連づけます。

実際のサービスを提供するクラスは、認証モデルから独立して拡張できるように、「具象サービス」として「サービス」クラスから派生させています。

認証していない段階のオブジェクト構成

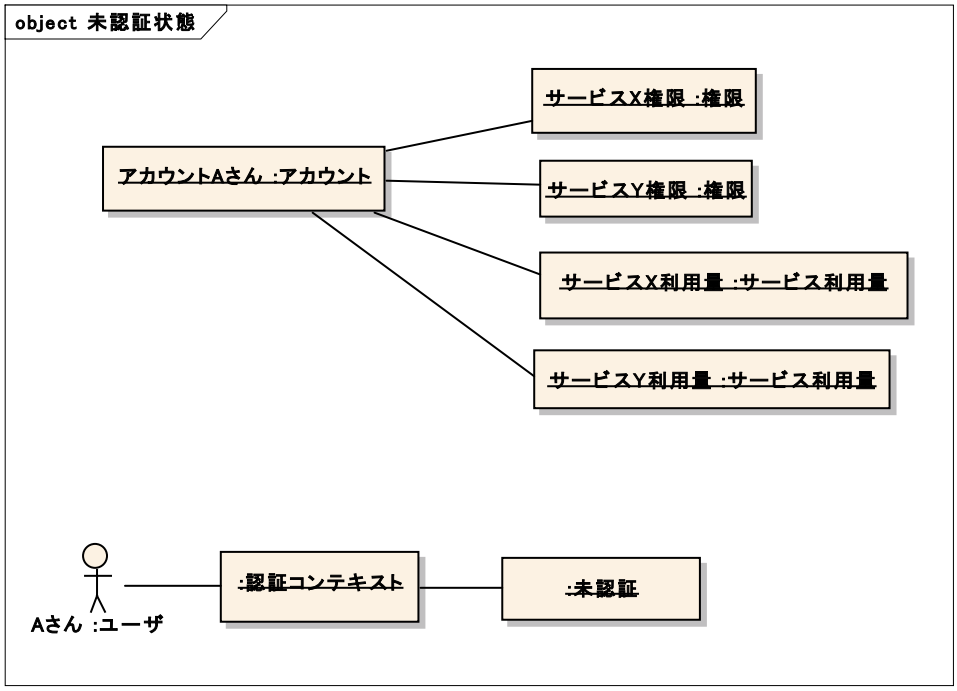


図24

このオブジェクト図では、1 つのアカウント「アカウント A さん」があり、2 つのサービス「サービス X」と「サービス Y」があるシステムを示したものです。

未認証状態では、ユーザ A さんはアカウントの情報を辿ることも、サービスを使うこともできません。

認証済みでサービスが利用されていない状態のオブジェクト構成

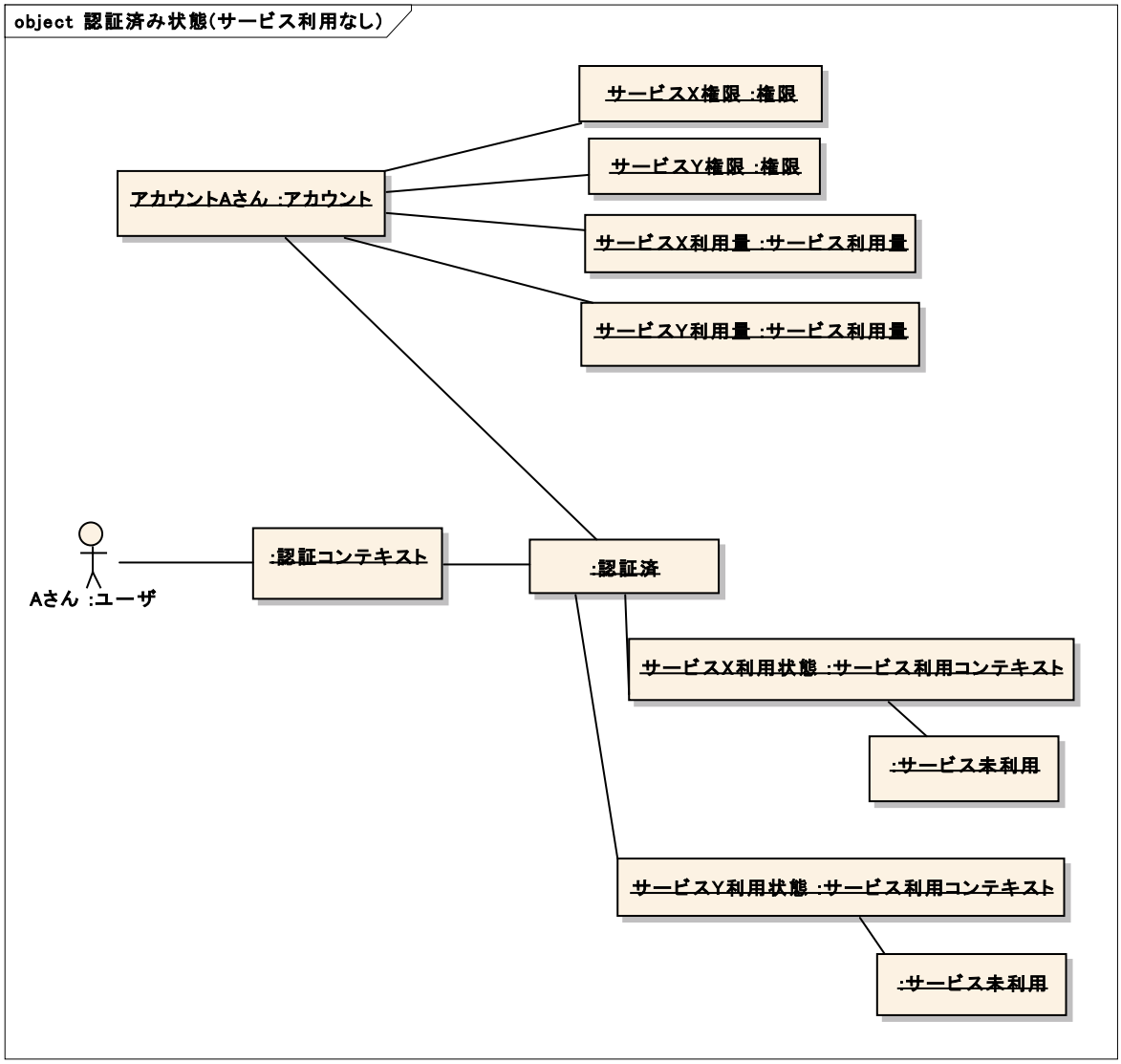


図25

認証状態では、「認証済み」のオブジェクトが「アカウント A さん」との関連を持ち、「ユーザ A さん」が、アカウント情報を辿れるようになります。

また、「サービス X」と「サービス Y」の利用コンテキストが生成され、サービスの利用を試みることができます。ただし、「サービス利用コンテキスト」には「サービス未利用」状態が関連付けられており、承認処理を経由しなければサービスを利用することができなくなっています。

認証済みでサービス X を利用中のオブジェクト構成

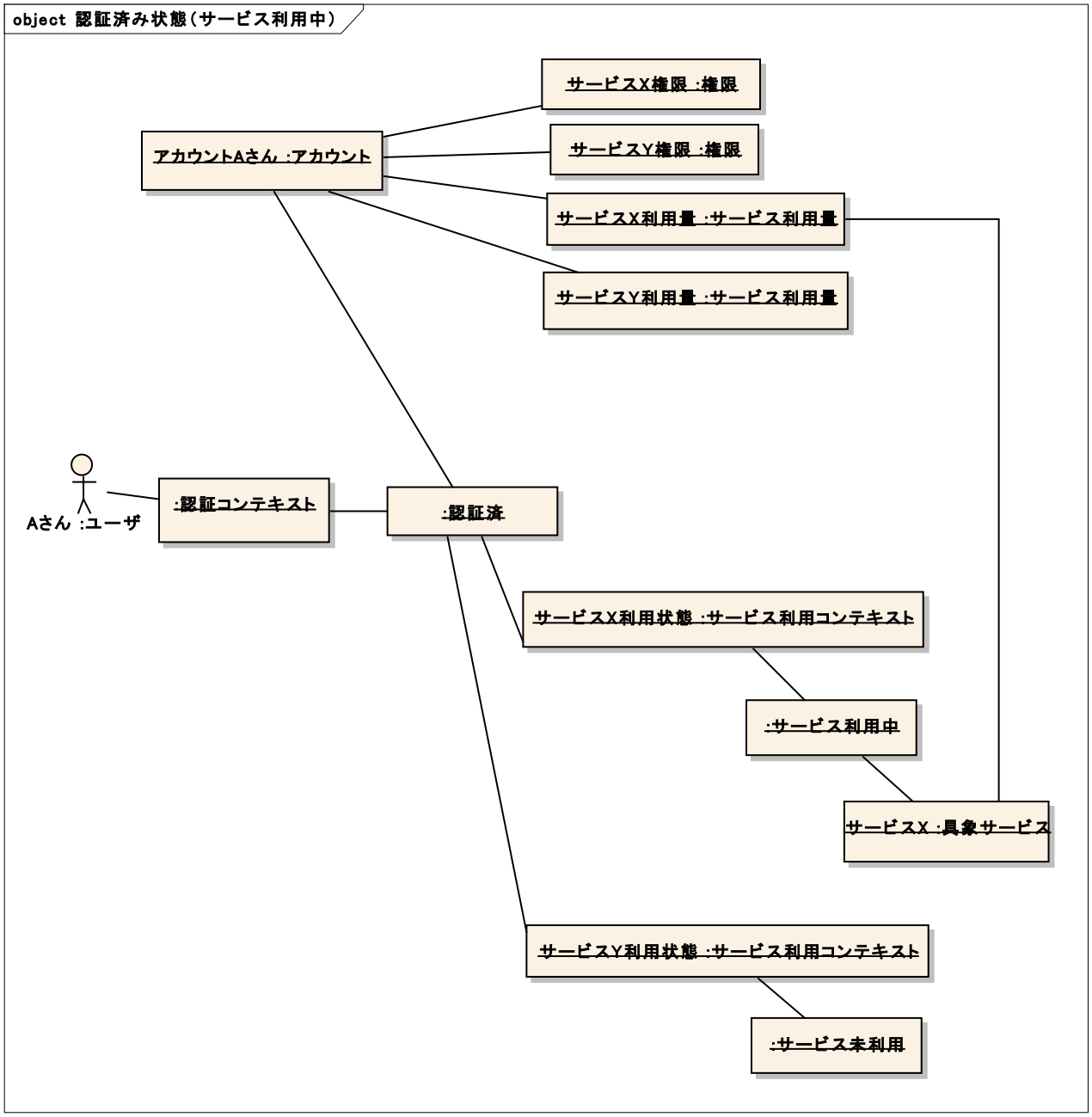


図26

この図は、「ユーザ A さん」が、「サービス X」を利用している状態を示しています。

サービス X のためのサービス利用コンテキスト「サービス X 利用状態」には、「サービス利用中」オブジェクトが関連付けられ、その先に「サービス X」オブジェクトが関連付けられています。また、「サービス X」は、「サービス X 利用量」との関連を持ち、適宜、サービス利用量を更新します。

ステートパターンを適用し、状態に応じたオブジェクトを組み合わせて認証モデルを実現します。

動的モデル

ユーザを認証するときの相互作用

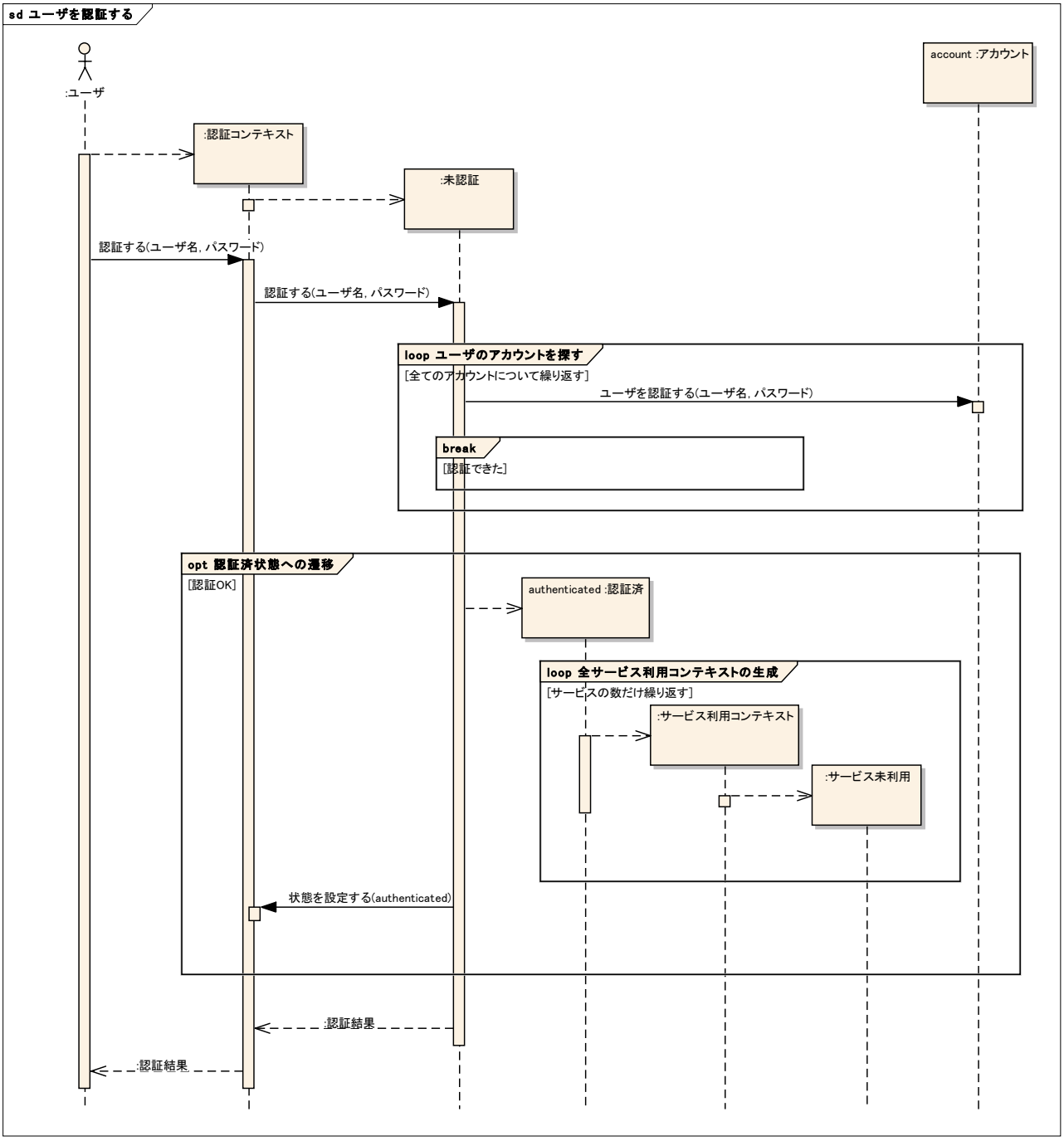


図27

サービスを利用するときの相互作用

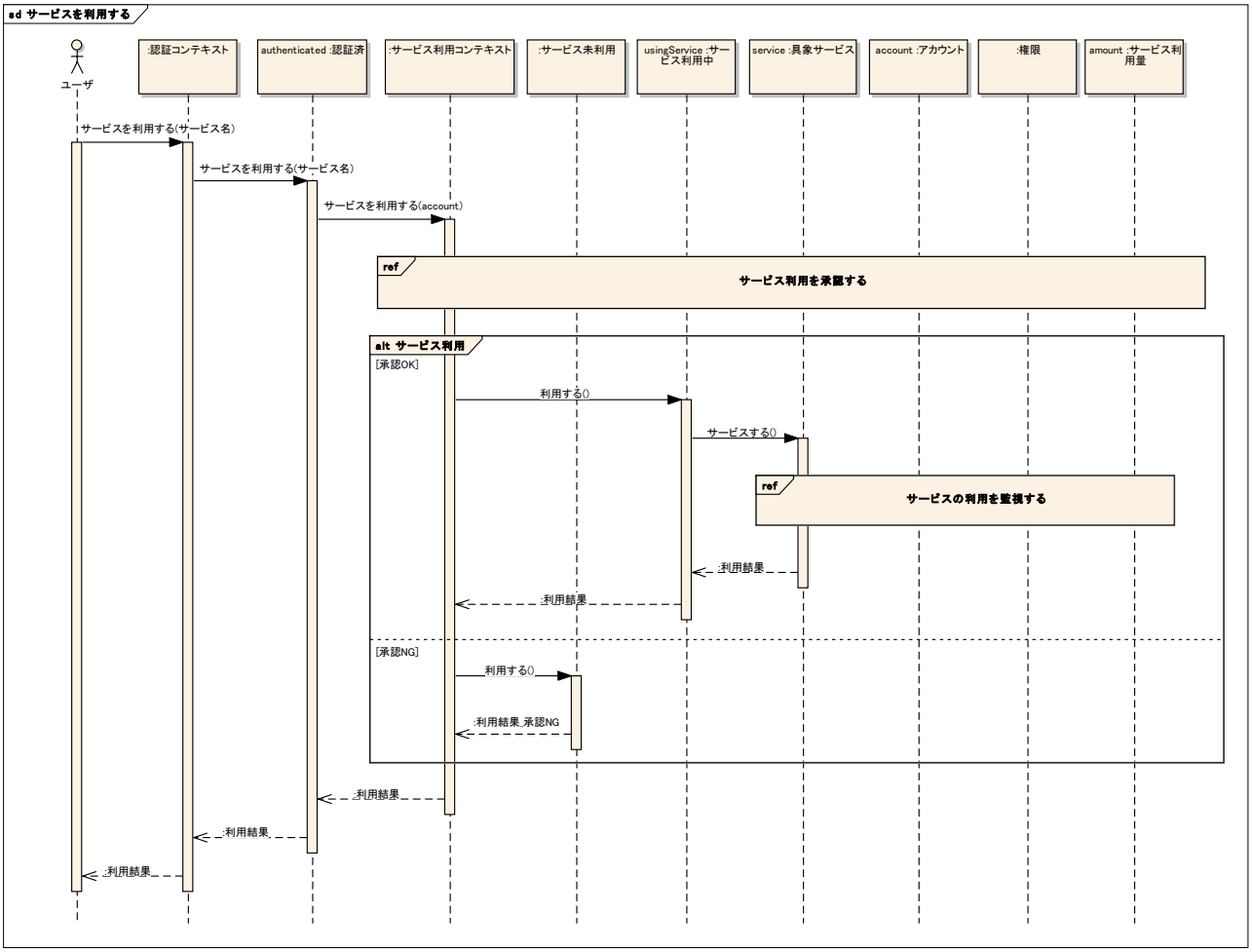


図28

サービスの実行を承認するときの相互作用

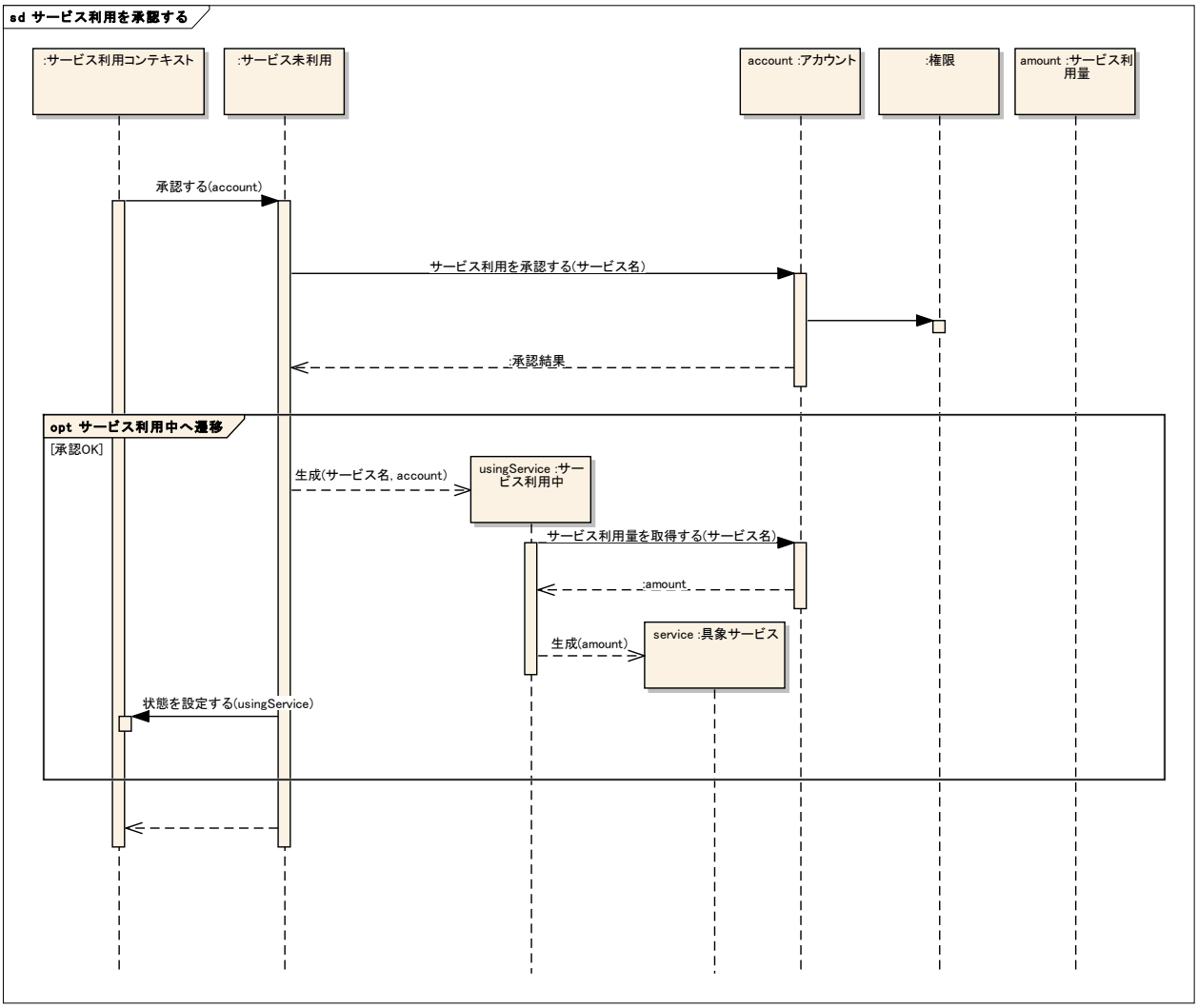


図29

サービス利用状況を監視するときの相互作用

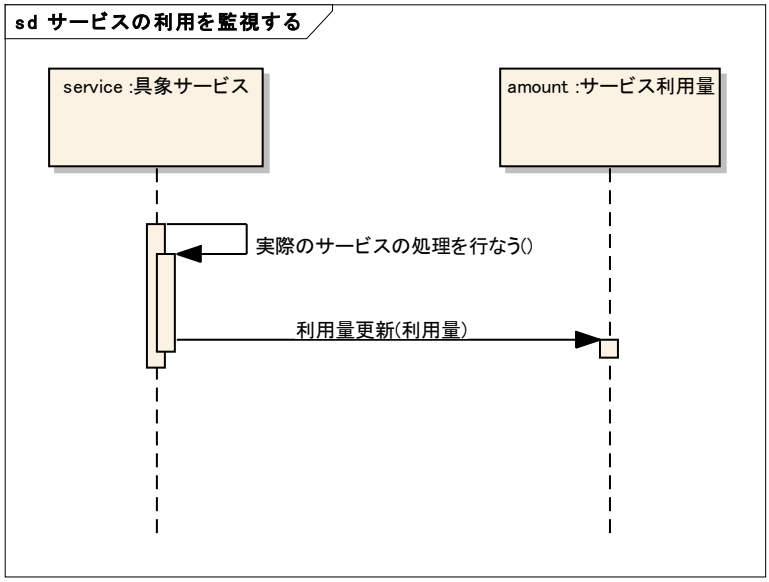


図30

組込み分野のための UML モデル解説書
「認証」

初版発行 ２０１０年（平成２２年）１０月１日
第二版発行 ２０１２年（平成２４年）５月１７日
発行者　UMTP, Japan
編　著　組込みモデリング部会
印刷

UMTP, Japan
東京都渋谷区代々木 1 丁目 22 番 1 号
http://www.umtp-japan.org/

Copyright(c)2010-2016 consortium for UML based Modeling Technologies Promotion All Rights Reserved