PlantUMLを通じてユースケース図の書き方を学ぶ

はじめに

ソフトウェアの仕様書、設計書の作成や管理を効率化するために、Markdown + PlantUMLによる作成方法を日々模索しています。
しかしながら、そもそもUML図の正式な書き方というものをちゃんと分かっていないというのが正直なところなので、PlantUMLを通じてUMLの各種図の書き方を勉強していきます。

- 本稿のテーマは、「UMLによるユースケース図の書き方」です。
- 本稿におけるUML図の作成は、Markdown + PlantUMLがベースであることを前提とします。

目次

ユースケース記述とユースケース図

- ソフトウェア開発の要件定義では、振る舞いモデルとドメインモデルの２つを柱として要件を分析・定義する。
- 振る舞いモデルは、ユースケースモデルにあたる。ユースケースとは、システムの利用例、つまりはユーザが利用できる具体的なサービスのことである。
- 通販サイトを例にすると、「商品をカートに入れる」「注文する」などがユースケースにあたる。
- 各ユースケースに対して、ユーザが何をしてシステムがどう振る舞うかをシナリオとして記述することをユースケース記述という。
- 記述したユースケースの中で、登場人物(システムやユーザなど)が達成する目的や範囲を図示したものがユースケース図である。

ユースケースを洗い出す際のポイント

- ユースケースを書き始めると粒度が分からなくなってしまうことが多い。
- 「○○が××を△△する」といったSVO形式で文を統一する。
- 事前に描いたドメインモデル図に登場する用語を使い、ユースケース図との間の整合性を取りやすくする。
- 具体的な表現を心掛ける。曖昧でぼやけていると、そのユースケースに対するテストシナリオも書きづらくなる。

基本コースと代替コース

- ユースケースの記述は、「基本コース」と「代替コース」の２パートから成る。
- 基本コース: 正常系のシナリオ文章
- 代替コース: 異常系・分岐系のシナリオ文章
- ユースケース記述では特に、代替コースをきちんと漏れなく書くことが重要になる。

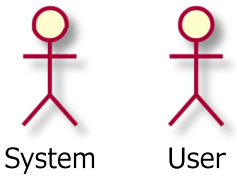
ユースケース図を構成する要素

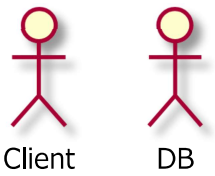
アクター

- ユーザやシステムの事を、アクターという登場人物として表現する。
- システムを利用する何らかの役割を持つオブジェクトだと考える。
- シナリオ文章の主語にあたる部分
- 人だろうがシステムだろうが棒人間で表現されることに注意

PlantUMLでの記述例

```
:System:
:User:
actor Client
actor DB
```



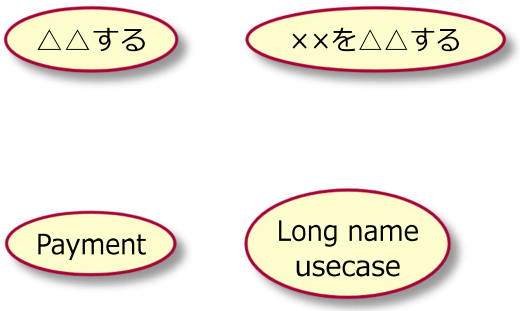


ユースケース

- ユースケースは楕円で表現される。細かい表現はさけて、△△するという動詞だけで表現する。
- 初めはシンプルにするよう心掛け、アクターやユースケースが出し切れたところで、「××を△△する」みたいに目的語を付けてブラッシュアップしていく。

PlantUMLでの記述例

```
(△△する)
(××を△△する)
usecase Payment
usecase (Long name \n usecase) as long
```

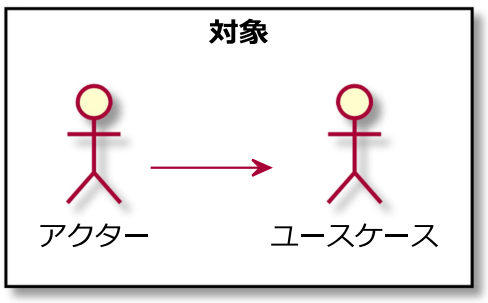


対象

- ユースケースが実現する対象を示し、長方形で表現される。
- システム境界線とも言う。
- どのシステムで実現するのかシステム範囲をしっかり表現するために、システム名称などのラベルを書くようにする。

PlantUMLでの記述例
対象の中にユースケースを含める形で描く

```
left to right direction
rectangle 対象 {
    actor アクター
    actor ユースケース
    アクター --> ユースケース
}
```

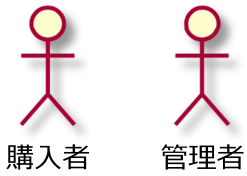


ユースケース図を描く流れ

ここでは、ECシステムを例としてユースケース図を描いてみる。

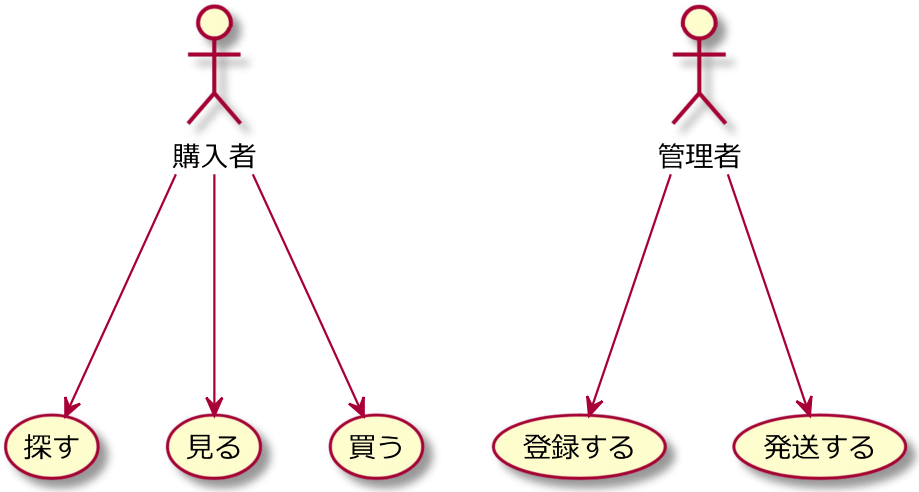
1. アクターを列挙する

- シンプルに考えると、登場するアクターは購入者と管理者の２種類。



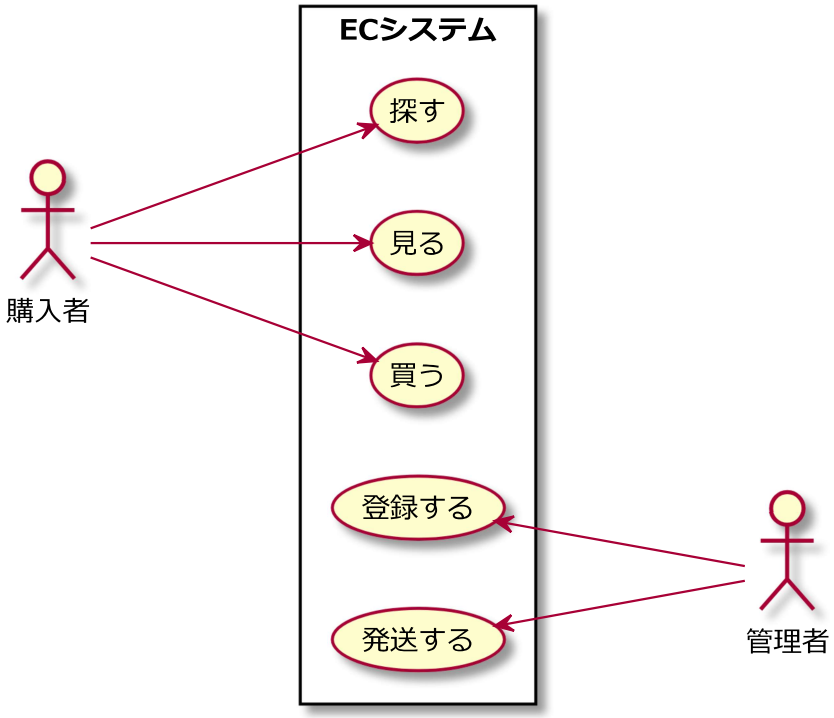
2. アクター毎のユースケースを列挙する

- 「○○が××する」という簡潔な形でユースケースを書く。
- この段階では、細かい事は一旦省いて書く。



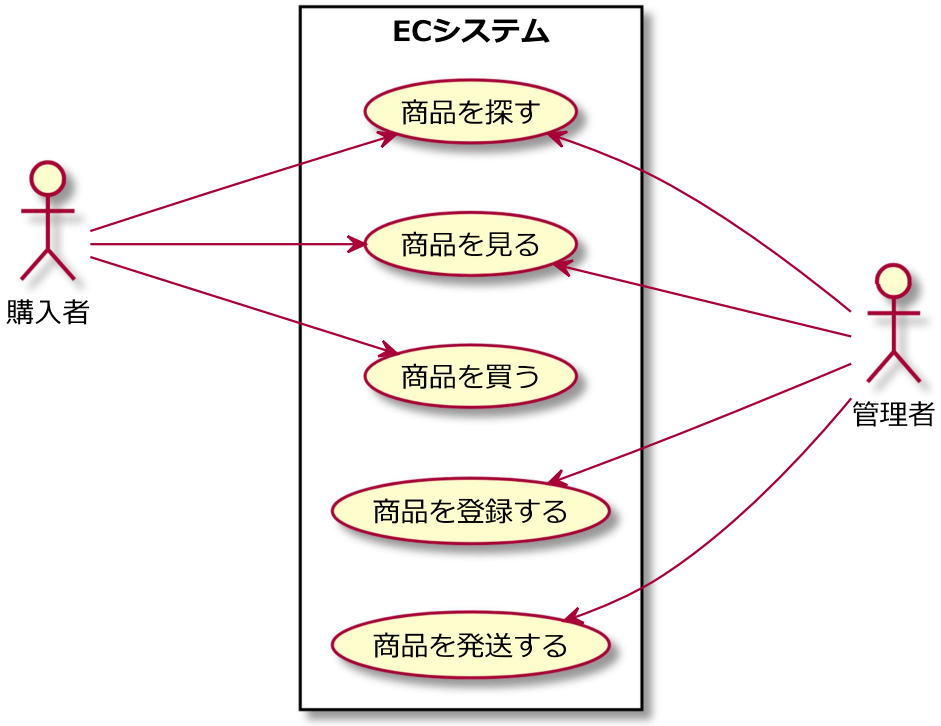
3. 対象に収める

- アクターとユースケースが列挙出来たら、対象に収める。
- アクター毎のユースケースが分かるように、配置などを整理する。



4. ユースケースを整理する

- STEP2で作ったユースケースの列挙を整理する。
- 「○○が△△を××する」という形に変更していく。
- 同じユースケースが他のアクターでも発生しないか考える。



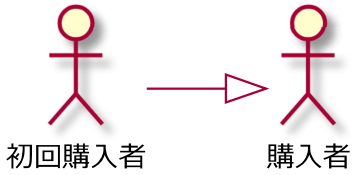
- アクターの定義や、対象システムが対応する範囲を明確にできる。
- より具体的なシーンとして理解できるように記述を添えると更に良い。

ユースケースの表現パターン

パターン1: 汎化(Generalization)

- クラス図における汎化と同じ意味。先から基へ矢印が向く。
- 「is a」関係、つまり「B is a A」が成り立つ。
- こういうアクターやユースケースも含まれていますよ的に使う。

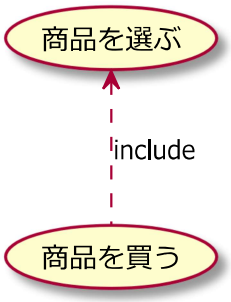
汎化の例



パターン2: 包含(Include)

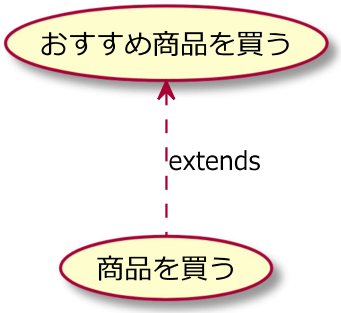
- 他のユースケースを含むと表現したい時に使う。
- 「has a」関係、「AはBを含む」が成り立つ。

「商品を買うは商品を選ぶを含む」を表現した例



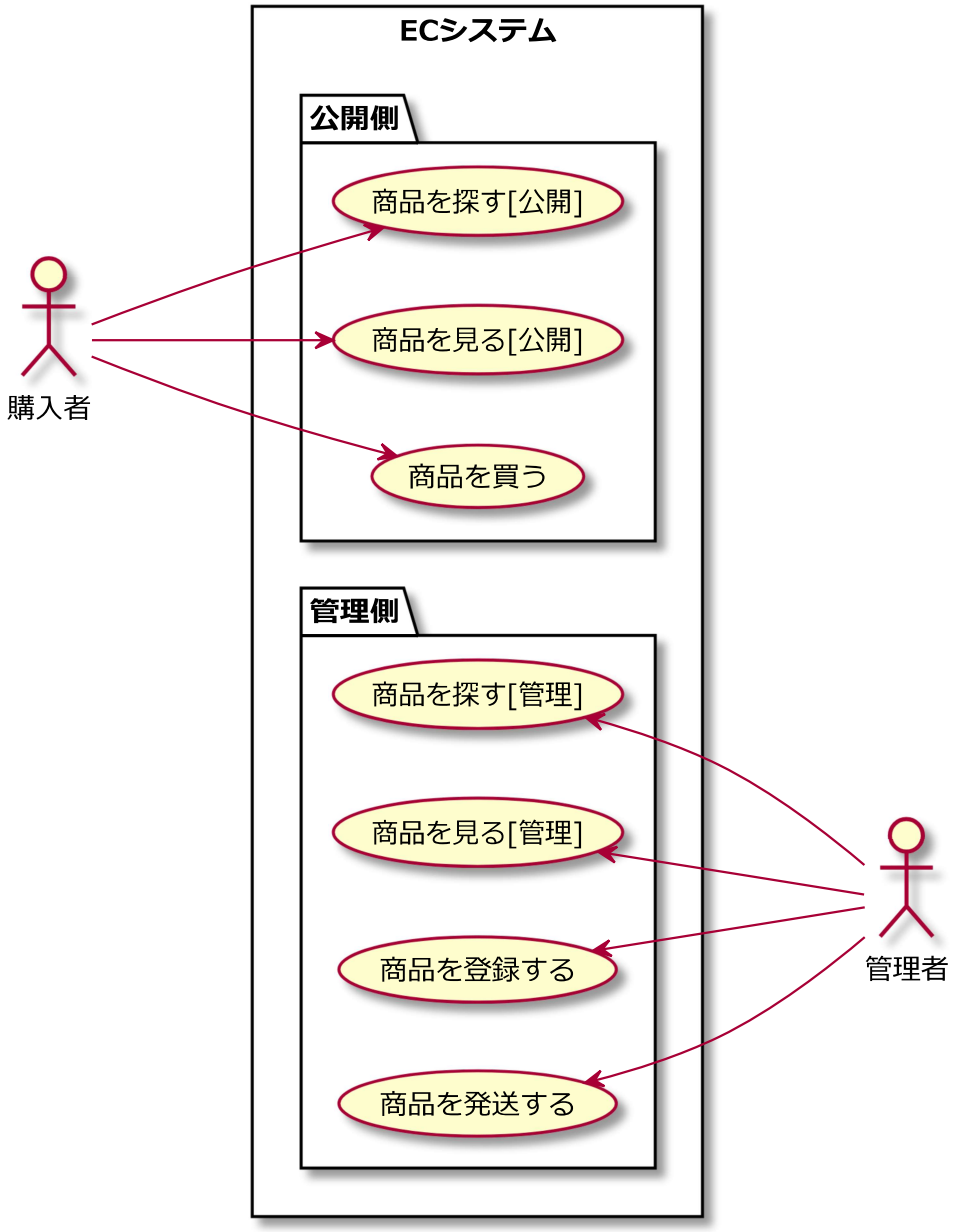
パターン3: 拡張(Extend)

- 包含とは異なり、ある条件の時だけ利用する場合に使用する。



パターン4: パッケージ

- 複数のユースケースをまとめて表現する、サブシステムにまとめる際に利用する。
- 例で取り上げたECシステムのユースケースでは、購入者が「商品を探す」のと、管理者が「商品を探す」のとでは、探す条件や求める内容が異なる。



参考資料

- Asial/Developers/Blog UMLを描こう
- 【新人教育 資料】第5章 UMLまでの道 ～ユースケース図の説明&書いてみよう編～
- プログラマーズ雑記帳 PlantUML - ユースケース図