

# Sphinx事例紹介 ～Slaterの場合～

---

2015/10/31

JUS Sphinxワークショップ@関西

# 自己紹介

- Twitter: @kk\_Ataka
- GitHub: gosyujin
- Sler勤務
  - Sphinxはプライベートで利用しつつ、仕事にも組み込めないか試行錯誤中



# アジェンダ

## I. Sphinx導入前

- 導入のための政治活動

## 2. Sphinx導入中、導入した後

- 導入するにあたっての壁
- 「納品」するには

## 3. Sphinx導入事例と課題

# Sphinx導入前

# はじめにSphinx(reST)の知名度調査

- 知名度0！
- Markdownもあんまり
- Wikiはかろうじて多少知られている

まずは政治から

# 競合ツールと比較！

→ 競合ツールと比較しての良さを調査

1. Office(Word, Excel)

2. Wiki, Markdown

※ MarkdownはJekyllやHugoなどMarkdownで記述できる  
静的サイトジェネレータを想定しています

# 比較1 Office

# Office 長所

- SI界のスタンダード
- **WYSIWYG**な操作
  - きめ細かいデザインが可能
  - 図やフローの挿入が容易

# Office 長所

→ ものすごく複雑な箇条書きが簡単(?)に作れる

1.1. 設計方針

1.2. 開発スケジュール

1.2.1. 要件定義 // => ネストしていく

1.2.1.1. xx機能 // => さらにネスト...

1.2.1.2. yy機能 // => 色々書いて...

1.2.2. 基本設計 // => ネストからの復帰

1.3. 開発体制 // => さらに復帰

# Office 長所

- エグい表/テーブルが簡単(?)に作れる
- 連結、結合たくさんあるマトリクスのようなものとか
- **Excel**方眼紙フォーマットで自由自在(泣)
- 値の計算が簡単
- これは表計算ソフト**Excel**の独壇場
- 表計算の用途に**Excel**を使うのは賛成

# Office 短所

- `diff`が取るのがメンドくさい
  - 取れないとは言ってない、最近は取れるっぽい
  - 往々にして日付バージョン管理とくっつく事が多い
  - VCSと相性悪い
- 恐怖の 「議事録\_20140505\_2(最新)(xx修正).xls」

# Office 短所

- 検索しづらい
  - 違うシートとか、吹出しどとか 非表示とか...探せない...
- ミリ単位のレイアウト修正を強いられる
  - 大事なのは内容...そうでしょ！
- 職人芸が発揮されるほど重い
  - 1GBオーバーのWordファイル...とか

# 比較2 Wiki, Markdown

# Wiki, Markdownの長所

Officeで短所として挙げた問題は解消できている！...と思う

# Wiki, Markdownの長所

"diffが取るのがめんどくさい"

- Markdownはプレーンテキストなので簡単
  - バージョン管理もしやすい
- Wikiもだいたい差分表示機能あり
  - diff取りやすい

# Wiki, Markdownの長所

## "検索しづらい"

- Officeよりは探しやすいんじゃないかと思うのだが...
- ブラウザ、エディタの検索とか、Wiki内検索とか

## "ミリ単位のレイアウト修正"

- 出力先(html+cssなど)である程度統一できる

# Wiki, Markdownの短所

Officeでは特に意識していなかったことを考慮する必要あり

# Wiki, Markdownの短所

- 記法を覚える必要がある
  - 未経験者に敷居が高い...
  - 図やフローは基本的にタグで挿入
- 「特定部分のみ」のレイアウト修正が面倒
  - 独自の処理を入れる？面倒...

# Wiki, Markdownの短所

- しっかり作らないと情報が散らかる
  - いわゆるTips集になってしまう
- 方言が多い(特にMarkdown)
  - PHP Markdown Extra, GitHub Flavored Markdown etc
  - パーサが異なると出力結果が変わってしまう
    - 逆に考えると表現方法が増えるため長所になり得る
- 出力はhtmlを想定しているものが多い

そしてSphinx

# Sphinxの長所

- Wiki, Markdownの長所と短所はだいたいSphinxにも当てはまる
- Sphinxが強いところは...

# Sphinxだけの長所

- 体系统的なドキュメント の骨組みを 簡単に 整えられる
- これだけで導入する価値あり
- 章の組み換え等も簡単
  - Wikiとかでこれを整備するのはちょっとしんどい...Office(Word)はちょっと得意かも
- 実際に作ってみると実感がわきづらいと思う

## 目次

[軽量マークアップ言語の歴史](#)  
[TODOリスト](#)  
[Indices and tables](#)

## 次のトピックへ

[1. はじめに](#)

## このページ

[ソースコードを表示](#)

## クイック検索

検索

モジュール、クラス、または関数名  
を入力してください

## 軽量マークアップ言語の歴史

### 目次

- [1. はじめに](#)
  - [1.1. 本書の目的とゴール](#)
  - [1.2. 注意事項](#)
  - [1.3. マークアップ言語](#)
  - [1.4. 軽量マークアップ言語](#)
  - [1.5. オレオレマークアップ](#)
  - [1.6. 2015年、存在感のある流派](#)
- [2. reStructuredTextの章](#)
  - [2.1. setext](#)
  - [2.2. StructuredText](#)
  - [2.3. reStructuredText](#)
- [3. Markdownの章](#)
  - [3.1. Markdown](#)
  - [3.2. CommonMark](#)
- [4. Wiki\(WikiWikiWeb\)の章](#)
  - [4.1. WikiWikiWeb](#)
  - [4.2. YukiWiki](#)
  - [4.3. MediaWiki](#)
- [5. その他の言語の章](#)
  - [5.1. Textile](#)
  - [5.2. はてな記法](#)
- [6. まとめ](#)
  - [6.1. 軽量マークアップ言語の年表](#)
  - [6.2. 終わりに](#)
  - [6.3. 本書の執筆環境](#)
- [付録: 参考](#)
  - [参考文献](#)
  - [参考サイト](#)
- [付録: 用語集](#)

# Sphinxだけの長所

- 複数ページをまたぐのも得意
  - 索引ページ、用語集ページの作成も簡単
- html以外にも出力形式が豊富
  - text, pdf, epub, etc...

# 索引ページ

→ こんな感じ

## 索引

[A](#) | [B](#) | [C](#) | [D](#) | [G](#) | [H](#) | [I](#) | [J](#) | [M](#) | [N](#) | [P](#) | [R](#) | [S](#) | [T](#) | [V](#) | [W](#) | [Y](#) | [記号](#)

### A

[Asciidoc](#)

### B

[Benjamin Dumke-von der Ehe](#)

### C

[CommonMark](#)

### D

[David Goodger](#)

[David Greenspan](#)

[Dean Allen](#)

[Doxygen](#)

### G

[GFM](#)

[GitHub Flavored Markdown](#)

### H

[HTML](#)

# 用語集ページ

## 付録: 用語集

- 用語一覧

→ こんな感じ

### 用語一覧

#### Asciidoc

[Stuart Rackham](#) によって2002/11/25([AsciiDoc ChangeLog](#))に作成された。Human Readableを目指した言語で、Pro Git 第2版は[Markdown](#) から全面的に[Asciidoc](#)へ書きなおされています。[テクニカルライティングの将来 – GitHub上のAsciidocで技術書Pro Gitを協働執筆 | 開発手法・プロジェクト管理 | POSTD](#)

#### Benjamin Dumke-von der Ehe

[CommonMark](#) 提唱者の一人。pagedown作者。Stack Overflow所属。

#### CommonMark

[John MacFarlane](#) を中心に立ち上げした[Markdown](#)共通化プロジェクト。2015年、[Markdown](#)の後継となるべく仕様策定中。

#### David Goodger

[reST](#) の作者。

#### David Greenspan

[CommonMark](#) 提唱者の一人。EtherPad、Meteor作者。

#### Dean Allen

[Textile](#) 作者。

#### Doxxygen

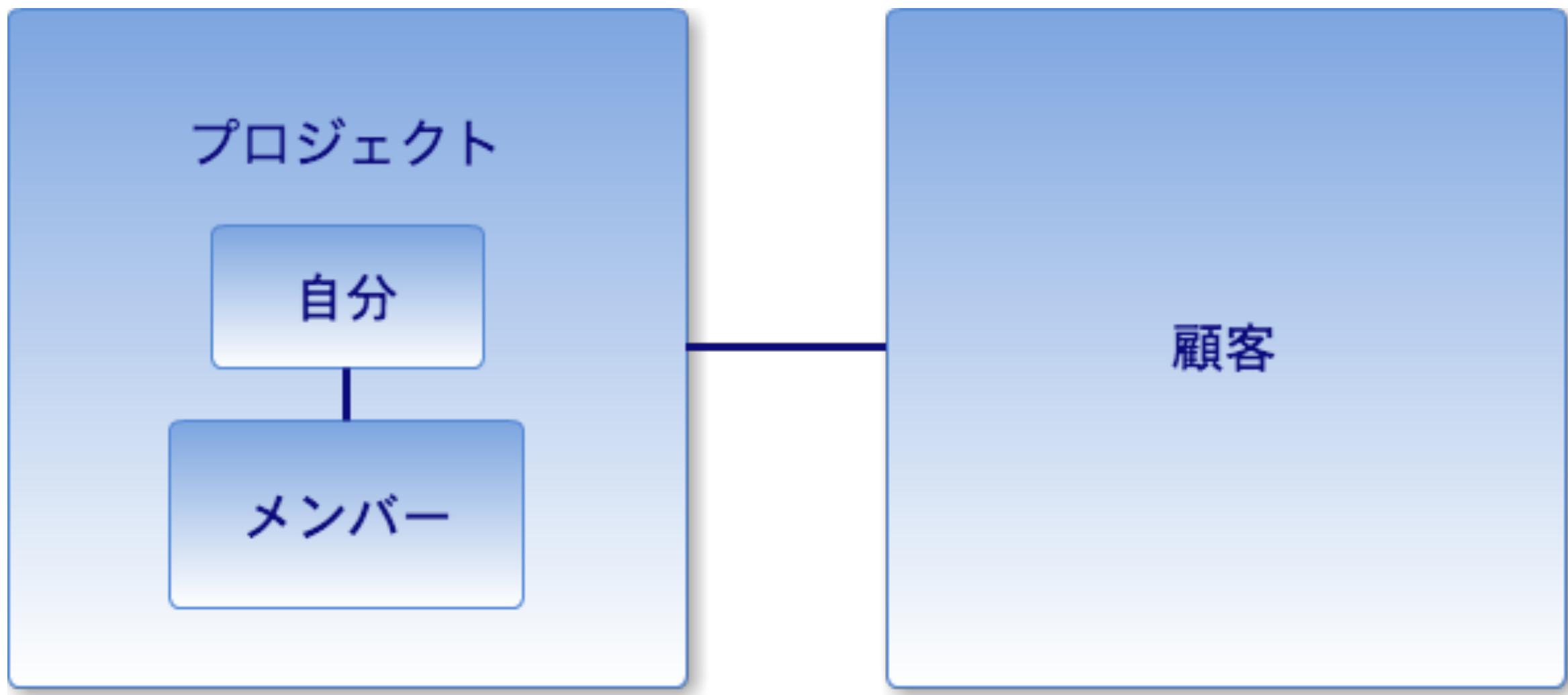
##### 課題

Doxxygenの一口メモ書く

Sphinx導入中  
と  
導入した後

導入するにあたっての壁

1. 対、プロジェクトのメンバー(PM)に対して(布教)
2. 対、顧客に対して(納品)



# 壁1. 対PM

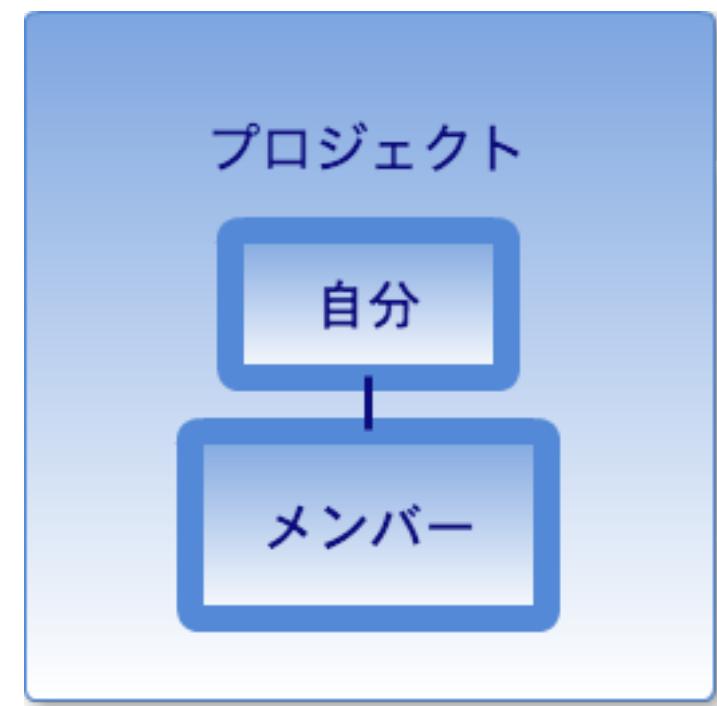
# 対PM 登場人物

## I. 自分

- Sphinxを導入したい人、基本的にはなんでもやる

## 2. メンバー

- 導入したSphinxを使ってほしい人



# 対PM 「自分」の仕事

- 今回はreSTで進めることの明確な宣言とサポート
  - 一番大事
  - これができないと負の成果物が生成される...

# 対PM 「自分」の仕事

→ メンバーが「rstファイルを開いてドキュメント作成」に注力できる環境を作る

1. `sphinx-quickstart`で下準備
2. ドキュメント自体のアウトライン作成
3. `doctree`の作成

などなど

# 対PM 「自分」の仕事

- ビルド環境、デプロイ環境などもお膳立て
- ビルドは**Jenkins**などで拾う
- デプロイは**Web**サーバにhtmlファイル配備とかがお手軽

# 対PM 「メンバー」の仕事

- とりあえずreST記法を覚えてもらう
- commitすれば(ビルド後)成果物が出来ることを周知

# 対PM 課題

- ローカルPCでのプレビュー
- メンバーのPCにSphinxを入れてもらうのは厳しい...
  - 確認できるのがcommitした時のみになってしまう
- ローカルで簡単にreSTプレビューできない
  - GitHubとか使えばある程度できるんだけど

# 対PM 課題

- プロジェクトの風土にあわせたカスタマイズが必要
  - こういうレイアウトがいい
  - こういうヘッダフッタがほしい
  - 社風にあわせて

# 壁2. 対顧客

# 対顧客 登場人物

## I. プロジェクト

→ **Sphinx**でドキュメント納品する側

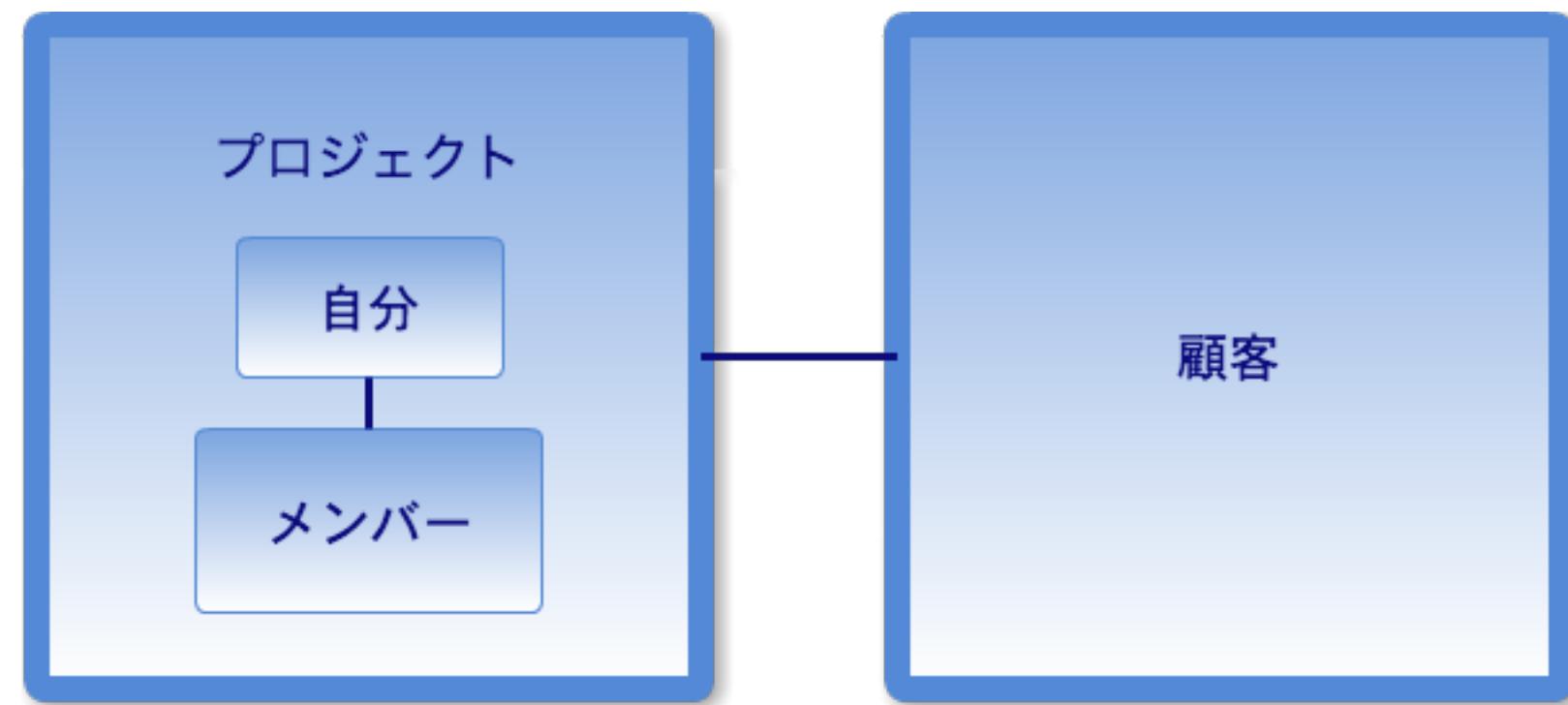
## 2. 顧客

→ ドキュメントを納品される側

→ 社内の人 or 社外の人

→ 歴史的経緯から**Office**で納品される事が多い

→ 例外は**Javadoc**とか？



# 対顧客 「プロジェクト」側の仕事

- 顧客に対して宣言&合意を得る
- 「今回はOfficeじゃない形式で設計書書きますよ」

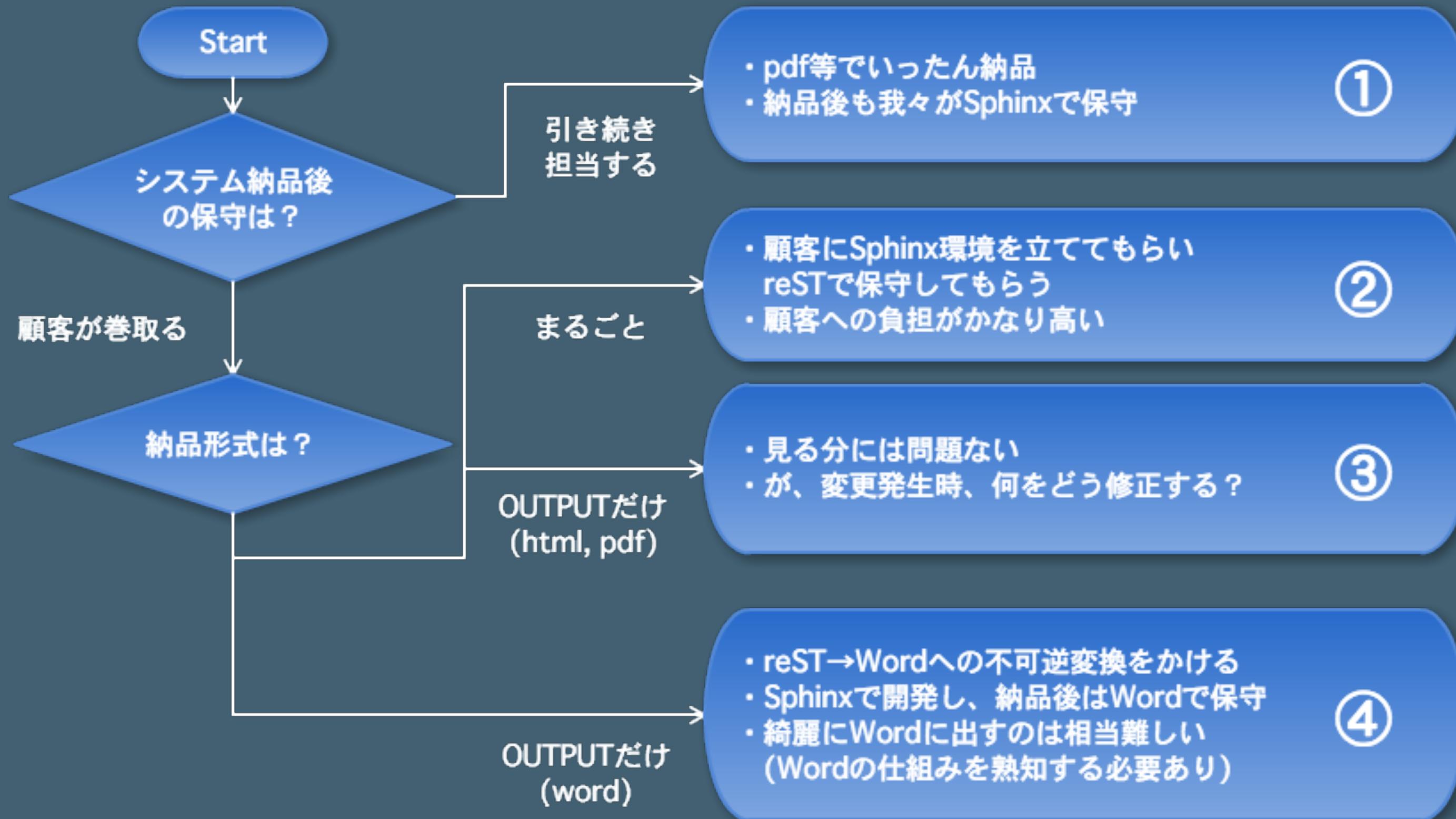
# 対顧客 「顧客」 側の仕事

→ 特になし？心構えくらい？

# 対顧客 課題

- 納品形式(次ページ参照)
  - 納品後 「誰が」 保守するのか
  - 顧客が巻取ってしまう場合、どう保守すればよいのか
    - 要解決事項
    - これがうまく解決できないと導入できない

「納品」するには…



自分たちで保守できない  
と厳しい...

# Sphinx導入事例と課題

# 環境

- 3ヶ月位のプロジェクトでSphinxを適用
- ほぼ1人で設計、製造、テストを担当
  - 途中で1人サポートに入ってくれた
- ドキュメント作成環境は Sphinx + Git(VCS) + Jenkins(Build)
- ターゲットは「社内資料」(顧客へ納品しない)

# 導入した感想

1. **Git**でバージョン管理、テキストなので差分管理も簡単！
2. 内容に集中できる！
3. お目当ての章が見やすい！探しやすい！
4. **Output**は、意外と営業の人を受けが良かった！  
→ 「え？ なにこれ？ なんてツール？ あとで教えて」

# Sphinx導入の課題

- I. 「Sphinxで書いていく！」 空気を作るのが難しい
  - ⇒ reSTで文章を書いてもらうのも難しい
  - ⇒ 「仲間」がいてくれると助かる
2. 「納品」するにはまだ課題が多い
  - ⇒ 解決の糸口は未だ見つからず...