演習コースⅠソフトウェア工学の基礎
2025年度 活動報告

演習コースⅠ研究員：

坂本 和司	横河電機株式会社　＜発表＞
江村 健志	アズビル株式会社
中西 日向子	アズビル株式会社
梁川 理沙	パナソニック株式会社 エレクトリックワークス社
平井 康貴	東芝システムテクノロジー株式会社
最首 友香子	株式会社日立ソリューションズ・クリエイト
星野 智彦	株式会社アイシン
山内 伽那子	日本電気株式会社

主査　　：猪塚 修
副主査　　：長谷川 祐一
アドバイザー：鷲崎 弘宜

一般財団法人日本科学技術連盟
第41年度ソフトウェア品質管理研究会
演習コースⅠ ソフトウェア工学の基礎
2026年3月13日

アジェンダ

- **演習コースＩの紹介**
  - コース概要、2025年度の演習テーマ
  - 気づき、本日の発表内容
- **現場で起きている悲劇を「お弁当」で考える**
  - 消えたお箸と幻のから揚げ
  - お弁当の例えで学ぶ要求工学理論
- **現場を守る3つのアクション**
  - アクション①：定期的なレビュー
  - アクション②：変更プロセスの明確化
  - アクション③：小さなマイルストーンの設定
- **最後に**

コース概要

＜演習コースⅠ ソフトウェア工学の基礎＞

- 代表的なソフトソフトウェア工学技術を「一通り」習得
- 産学両面に通じたその道の「第一人者」の講師陣による徹底指導
- とにかく実際に「やってみる」ことで深く理解し記憶
- 組織を超えた「仲間作り」と情報交換

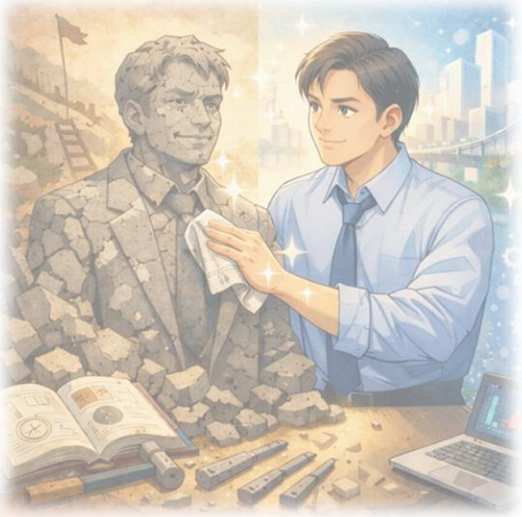
2025年度の演習テーマ

演習テーマ	講師
レビュー	猪塚 修 氏（横河ソリューションサービス株式会社）
オブジェクト指向分析設計	井上 樹 氏（株式会社豆蔵）
アーキテクチャ設計・評価	長谷川 裕一 氏（合同会社Starlight & Storm）
メトリクス	小笠原 氏（千葉工業大学情報変革科学部 認知情報科学科）
要求工学（要求分析）	斎藤 忍 氏（NTT株式会社）
アジャイル開発の基礎知識	天野 勝 氏（株式会社永和システムマネジメント）
ソフトウェアテスト	樋下田 順也 氏（株式会社ベリサーブ）
上流から下流まで生成AIが変革するシステム開発	酒匂 寛 氏（有限会社デザイナーズデン）
工数見積モデルの構築手法（CoBRA 法）	石谷 靖 氏（株式会社三菱総合研究所）
ゴール指向分析とGQM	鷲崎 弘宜 氏（早稲田大学グローバルソフトウェアエンジニアリング研究所/ 国立情報学研究所/ 株式会システム情報/ 株式会社エクスモーション）

気づき

- **社外の知見に触れて気づいたこと**
  - 自己流の知識と、体系的な基礎との「大きなギャップ」
  - ソフトウェア開発の基本を「分かっているつもり」だった
  - 開発現場で感じていることを言語化し、社外の知見からフィードバックを受けることは刺激となり、多くの学びを得られる

- **現場での経験を、ソフトウェア工学という理論の視点で磨き直し、再構造化するきっかけになった**



本日の発表内容

- **要求工学の視点から現場の問題を分析する、という気づき**
  - 現場では実際に何が起きているのか
  - それを要求工学の観点からどのように分析できるのか
  - どのような対処が考えられるのか

- **発表１：現場で起きている悲劇を「お弁当」で考える**
  - 動画解説

- **発表２：現場を守る3つのアクション**

現場で起きている悲劇を「お弁当」で考える

「明日お弁当を作ってほしい」と頼み、翌日お弁当を受け取った。

しかし、いざ食べようとした時、ソフトウェア開発における2つの「悲劇」が起こる。

動画の内容の補助資料（NotebookLMで作成）

現場で起きている悲劇を「お弁当」で考える

悲劇①：お箸がない＝「要求の抜け漏れ」

暗黙の前提

- お弁当は完成しているが、食べられない。
- 依頼者は「食べられる状態」が当然含まれていると期待していた。
- しかし、「お箸が必要」とは明示されていなかった。

ソフトウェア開発において、顧客の頭の中にある「当然」を明文化しないと、致命的な欠陥となる。

動画の内容の補助資料（NotebookLMで作成）

現場で起きている悲劇を「お弁当」で考える

悲劇②：やっぱり唐揚げが欲しい＝「スコープクリープ」

開発終盤での要求変更

- 「唐揚げはいらない」と合意していたはずが、後から追加を要求される。
- すでに詰め込まれたお弁当のバランス（設計）が崩壊する。

スコープクリープは、テストや実装の段階で予期せぬ設計のやり直しを強制する。

動画の内容の補助資料（NotebookLMで作成）

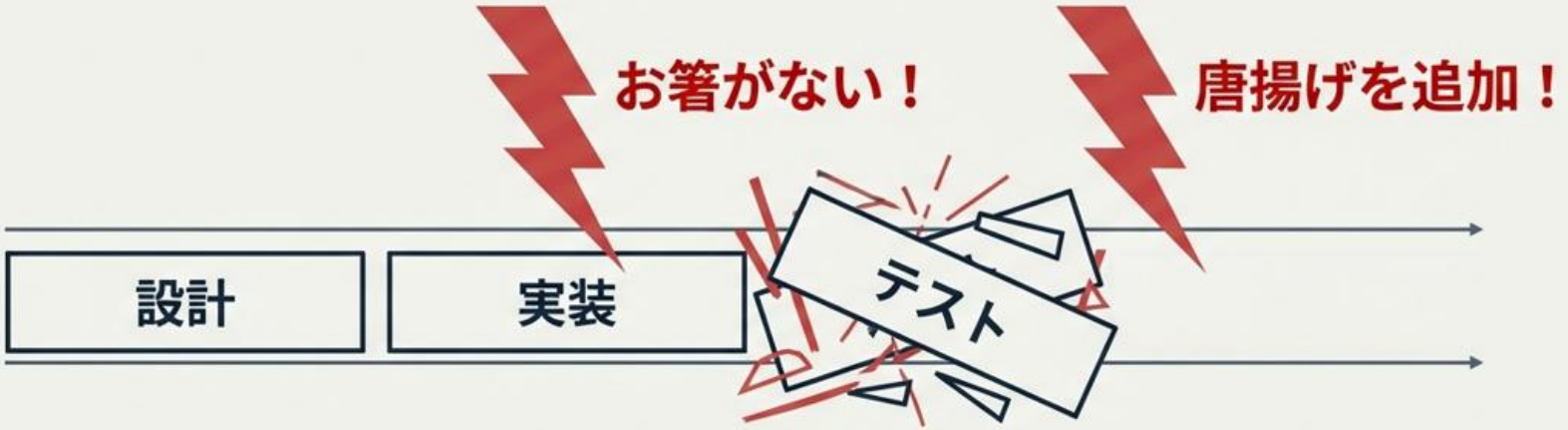
現場で起きている悲劇を「お弁当」で考える

現象	お弁当の例	理論的な正体
抜け漏れ	お箸がない	ドメイン知識（K）の定義不足
クリープ	やっぱり唐揚げ	要求（R）の不安定な変化

要求の抜け漏れや、要求クリープは必要だからおこる。
悪ではない。完全にはなくせない。

現実のプロジェクトで起きていること

要求の抜け漏れや、要求クリープに無防備だと・・・



- 固定された納期と予算
- テストと実装の並行による混乱
- リソースの枯渇とチームの疲弊

曖昧な要求と変更の連鎖を断ち切る「仕組み」が必要。

動画の内容の補助資料（NotebookLMで作成）

現場を守る「3つのアクション」

- **アクション①：定期的な全体レビュー**
- **アクション②：要求変更プロセスの明確化**
- **アクション③：小さなマイルストーンの設定**

属人的な努力ではなく、要求の抜け漏れとスコープクリープを防ぐための「構造」をプロジェクトに組み込む

アクション①：定期的な全体レビュー

- **「要求漏れ」を早期発見。全体の方向性を修正する**
  - 月に一度、ステークホルダーを交えて実施する。
  - 判断を下せる立場のプロダクトオーナーは必須。
  - お箸がない、のような「暗黙の前提の欠落」を早期に発見する
  - 「妥当性確認 (Validation)」のプロセスを組織的に実行する

やるべきこと	やってはいけないこと
・事実ベースでの進捗確認 ・課題を具体的にあげる ・参加者との合意 ・暗黙の前提の欠落がないか確認 ・建設的なフィードバックの要求	・あいまいな進捗報告 ・開発メンバだけでの実施 ・責任者不在、責任を曖昧にする ・完成してから見せる ・個人の責任追及、感情的な批判

「状況が見えない状態」から、
「全体像が共有され、優先順位が再確認された状態」へ変える会議

アクション②：要求変更プロセスの明確化

- **「要求クリープ」の抑制**
  - 変更は必ず記録。影響範囲を評価し、承認を通す仕組みを作る。
  - 唐揚げの追加を無秩序に受け入れないゲートキーパー。
  - なぜこの変更が必要になったのか、理由（ドメイン知識の変化）を数値とともに残す。

やるべきこと	やってはいけないこと
・変更の受付基準の明確化 ・記録、承認の徹底 ・影響評価(スケジュール、設計、etc.) ・要求・仕様ドキュメントを即座に更新	・声の大きさで決める ・口頭での曖昧な変更受け入れ ・評価せず開始し、失敗を繰り返す ・ドキュメントと実装の乖離

変更履歴は「報告用」ではなく「防具」
管理プロセスは官僚的なオーバヘッドではなく、
開発を苦しくしないための武器と考える

アクション③：小さなマイルストーンの設定

- **開発リズムの形成**
  - 短いサイクルで具体的な進捗を測り、小さなズレを即座に修正する「正当性検証 (Verification)」を繰り返し、開発を安定させる
    - ドメイン知識の欠落を早期に露呈させ、不確実性の解消
      - 外部環境や3rdパーティ製のソフトの振る舞いの誤認、など
    - 新たな要求が出てきた際、次の小さなマイルストーンへの影響を即座に評価できる

やるべきこと	やってはいけないこと
・担当者から具体的な成果・問題報告 ・優先順位の迅速な微調整 ・現実的な改善策の提示	・「なんとなく順調です」 ・スケジュールの遅れを放置 ・問題を各担当者に持ち帰らせる ・結局「がんばれ」という精神論のみ

「なんとなく進んでいる状態」を、
「具体的に前進している状態」へ変える会議

「3つのアクション」まとめ

アクション	具体的な行動内容	期待される効果	現場が気づくべきギャップ
定期的な全体レビュー	月に一度、ステークホルダーを交えて事実ベースで進捗と方向性を確認する。	**要求漏れの早期発見** 暗黙の前提を可視化し、軌道修正する。	• 「単なる報告の場」になっていないか？ • 「完成するまで見せない」が手戻りを生んでいないか？
要求変更プロセスの明確化	変更受付基準を決め、理由（ドメイン知識の変化）と影響範囲を必ず記録する。	**要求クリープの抑制** 記録を納期調整やリソース交渉のための「自分たちの武器」にする。	• 声の大きい人の一言でプロセスを無視していないか？ • 変更履歴がないため同じ失敗を繰り返していないか？
小さなマイルストーンの設定	1～2週間単位の短い間隔で、具体的な成果（仕様や実装の一部）を検証する。	**開発リズムの形成** 不確実な部分を早期に解消し、チームのストレスを軽減する。	• 数ヶ月先の大きな納期に向かって、不確実性を抱えたまま走り続けていないか？ • リスクを先送りしていないか？

終わりに

- **今回の活動を通して感じたこと**
  - 理論を知ることで、見過ごしていた論理的な破綻や構造的な問題に気づけるようになる
  - その結果、いまの頑張り方の延長では解決できない問題があると気づけるようになる
  - 違う頑張り方（考え方や構造を変える努力）に目が行くようになる