ソフトウェア工学
第13回

土田　隼之

週	授業内容・方法	週ごとの到達目標
1週	ソフトウェアの性質と開発	ソフトウェア開発の特徴および課題について少なくとも一つ上げられ、その理由を言える。
2週	ソフトウェア開発プロセス	複数の開発プロセスモデルを挙げ、それぞれの特徴を言える。
3週	要求分析	要求分析とプロトタイピングの関係性や有用性について言える。
4週	ソフトウェア設計	モジュールの結合度の低い場合と高い場合のモジュール間の依存性について述べ、モジュール結合度の低い具体例を言える。
5週	プログラミングとテスト	誤り混入をさせないためのプログラミング手法およびテスト効率を向上させる技法について言える。
6週	テストと保守	保守容易性を確保するための方策について、考察し、述べることができる。
7週	グループワーク	前半6週に関する課題を、グループワークで取り組む。
8週	中間試験	前半に習得した項目について確認する。
9週	オブジェクト指向１	身の回りのモノに関して、クラスとインスタンスという言葉を用いて説明できる。
10週	オブジェクト指向２	オブジェクト指向プログラミングの特徴について言える。
11週	ソフトウェア再利用	ソフトウェア再利用の重要性とその困難さについて言える。
12週	プロジェクト管理	プロジェクト管理の重要性を述べることができる。
13週	品質管理	品質管理手法について言える。
14週	ソフトウェア開発規模と見積もり	ソフトウェア開発規模の見積もり手法について言える。
15週	グループワーク	後半6週に関する課題を、グループワークで取り組む。
16週	期末試験	後半に習得した項目について確認する。

模擬試験を予定してます
#中間前の模擬試験同様に
スライド見ながら解いて良い

今日の内容

1)プロジェクト管理(WBSなど)の続き

2)品質管理

3)ソフトウェア開発規模と見積り

WBSの作成

WBSでは、必要な成果物を作るために実行する作業を階層的に要素分解する。

①トップダウンで作業を分解ください。

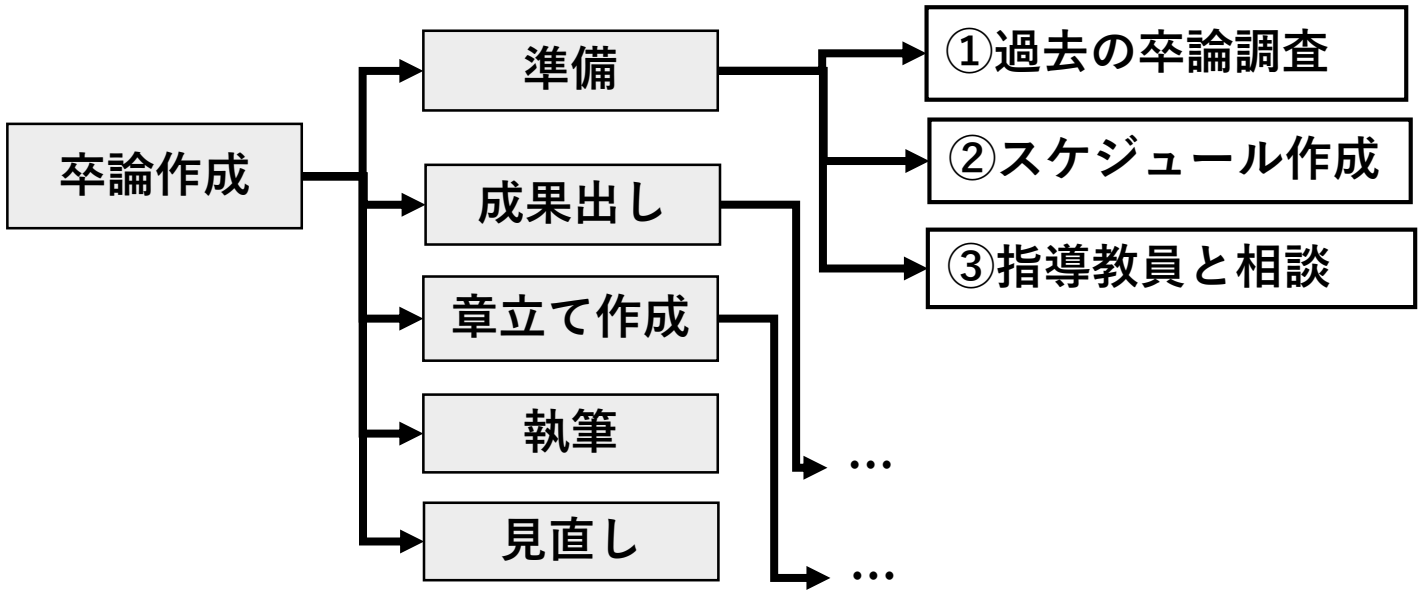
　全体から部分へ。最上位のWBSを定義して、上位から要素分解する。

②成果物ベース、プロセスベースの作業洗い出しを組合せる

　作成方針1:成果物ベース

例:在庫管理システム→サーバ、ネットワーク、プログラム→各構成設計の仕様

　作成方針2:プロセスベース:時間軸(作業の進め方) 例:基本設計、詳細設計



課題12-2:締切7/25

1)「グループワークで考案した案」のWBSをまずは個人で作成ください。各作業にかかりそうな時間も記載ください。

WBSは「案の仕様書作成」でも良いし、「案の開発」でも良いし、「案の教材作成」でも良いです。

#「数時間～半日以上くらいかかる作業」を最小単位としてください。

#細かすぎる作業は書かなくて良いです。

2)個人で作成したWBSをグループワーク班で共有して、考慮漏れが無いか確認ください。

グループワークですが、下記を各人で提出ください。

提出1)個人で作成したWBS(考慮漏れなどあれば修正後)を提出ください

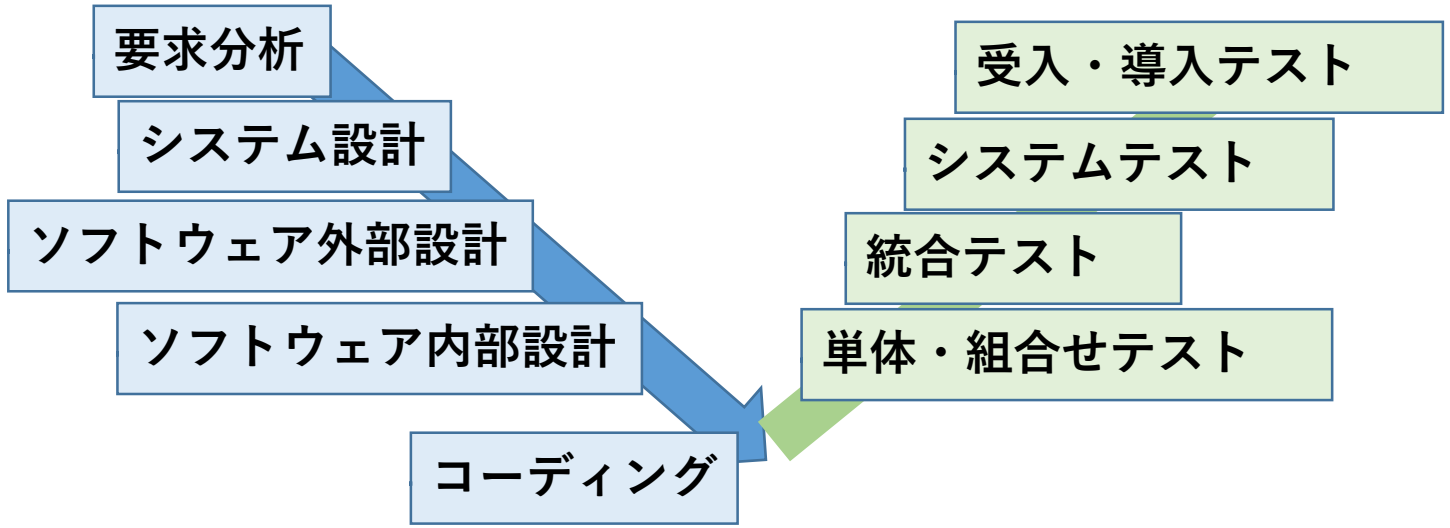
2)グループワーク班で共有した自分や相手の感想を提出ください。

第十五回の模擬試験の後に、
各班のWBSを相互評価してもらうつもりでいます
#評価基準:考慮漏れが無いか、現実的な見積りか

プロセス定義

開発作業を効率的に遂行し、管理するための作業の骨組みを定めることをプロセス定義という。プロセスは、大まかな開発段階(フェーズ)分けと各段階での成果物が設定されている。

通常、各企業は組織内にいくつか標準プロセスを持っており、開発するソフトウェアの性格に応じて仕立てて使用することが多い。例えば、ウォータフォールモデルはよく使われるモデルの一つである。管理プロセス例としては、設計と試験を対応付けたV字型モデルやスパイラルモデルがある。



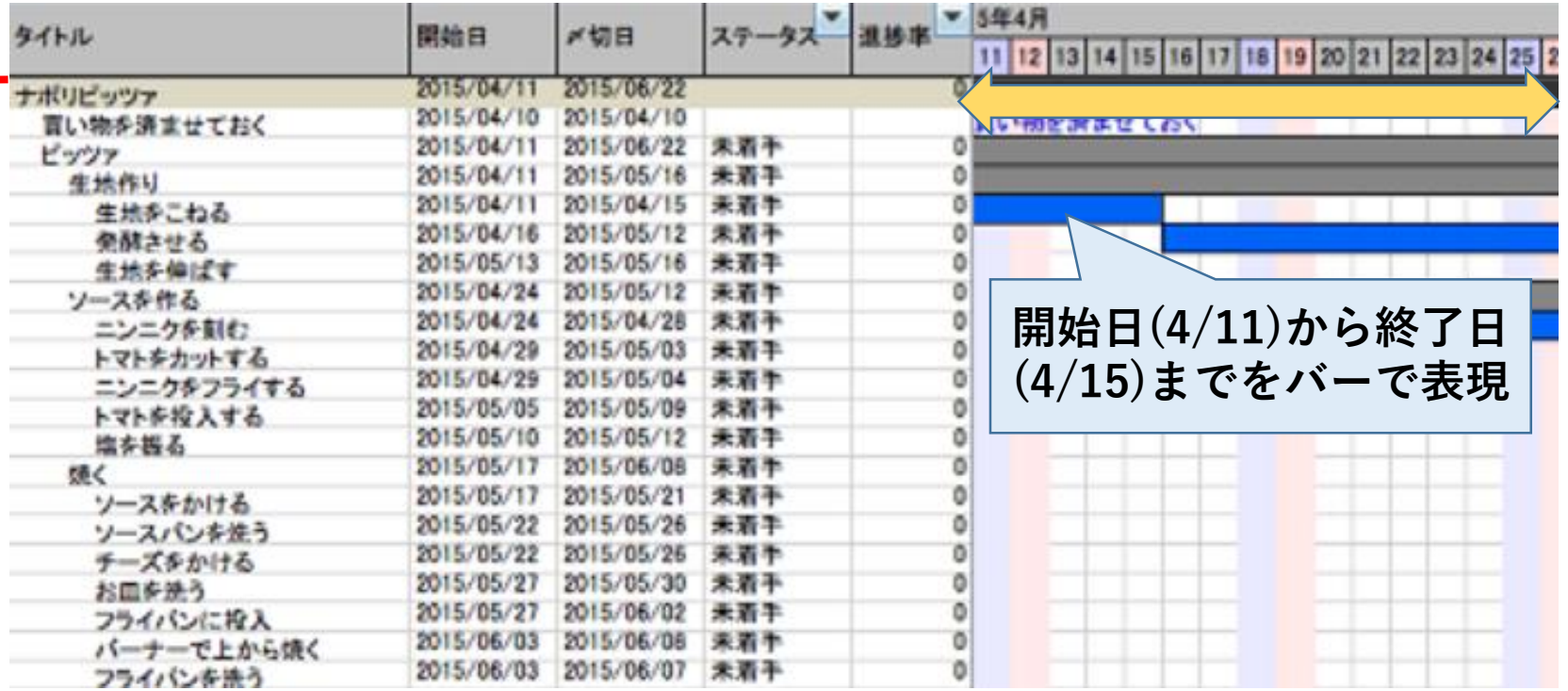
進捗管理:スケジュール計画1

スケジュール計画:作業リストと各作業の工数見積り、および各作業を担当する要因の関係を記述した責任分担表を入力と、各作業の開始日と終了日を設定する作業のこと。ガントチャートが簡便な手法としてよく使われる。

ガントチャート:縦軸に作業項目、横軸にカレンダー

各作業の開始日から終了日までをバーで表現

バーを塗りつぶして作業計画に対する実績分を割合で表現

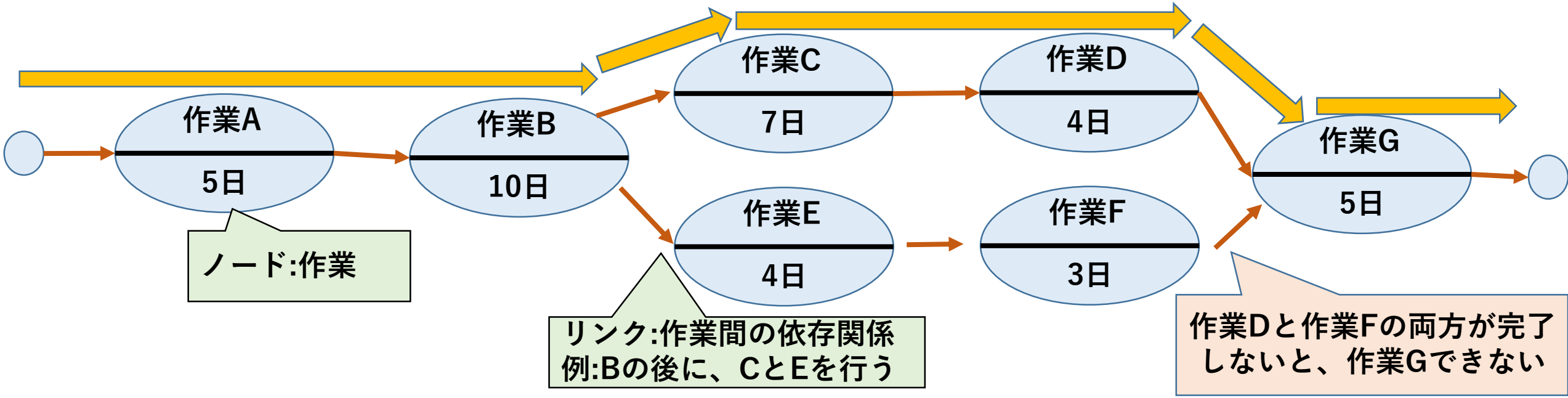


https://image.itmedia.co.jp/l/im/ait/articles/1504/24/l_gantt01_a.png#_ga=2.232284570.468402460.1643333675-1380744308.1611302470

進捗管理:スケジュール計画2

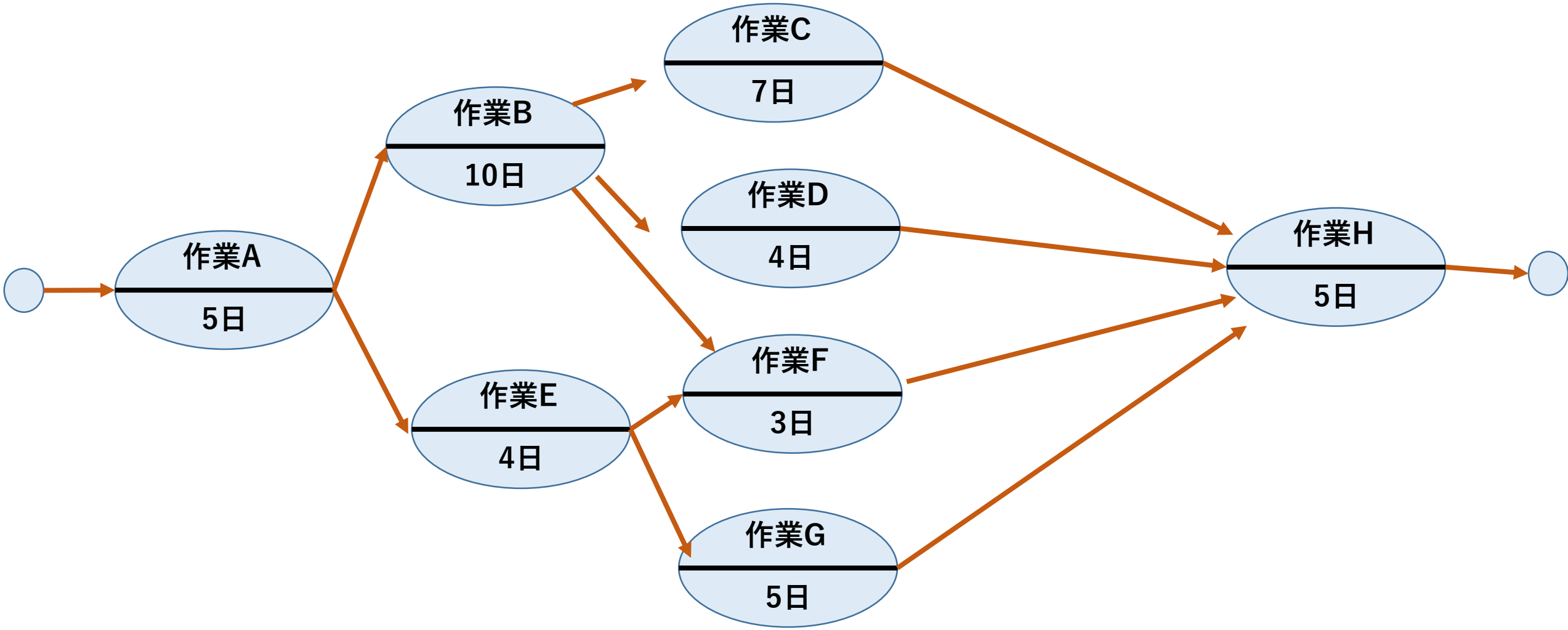
クリティカルパス法:各作業をノード、作業間の依存関係をリンクとする作業ネットワーク図を構成し、各作業に所要時間を記述することで、終了日や終了日から遡って開始日を計算する数理的な方法

クリティカルパス:最長の作業パスのこと。下記ではA→B→C→D→Gの31日となる。各作業の所要日数が短縮されなければ、31日が最短時間となる。



課題13-1:締切7/28

以下のネットワーク図のクリティカルパスとクリティカルパス期間を求めてください。



課題12-3:締切7/25

課題12-2で作成したWBSのワークパッケージについて、作業ネットワーク図を作成し、おおまかな作業日数を記載した後に、クリティカルパスを求めてみてください。

第十五回の模擬試験の後に、
各班のWBSを相互評価してもらうつもりでいます
#評価基準:考慮漏れが無いか、現実的な見積りか

今日の内容

1)プロジェクト管理(WBSなど)の続き

2)品質管理

3)ソフトウェア開発規模と見積り

品質管理

品質管理:ソフトウェア製品の品質を高い状態に保つために行う活動

①プロダクトの品質を直接管理する方法と、**②プロダクト開発のプロセスを保証・改善することでプロダクトの品質を間接的に管理する方法**がある。

ソフトウェアの品質:利用者にとって望ましいソフトウェアの性質

#例)機能完全性(業務を実現するために必要な機能を網羅していること)

時間効率性(実行速度やスループットが要求をみたすか)

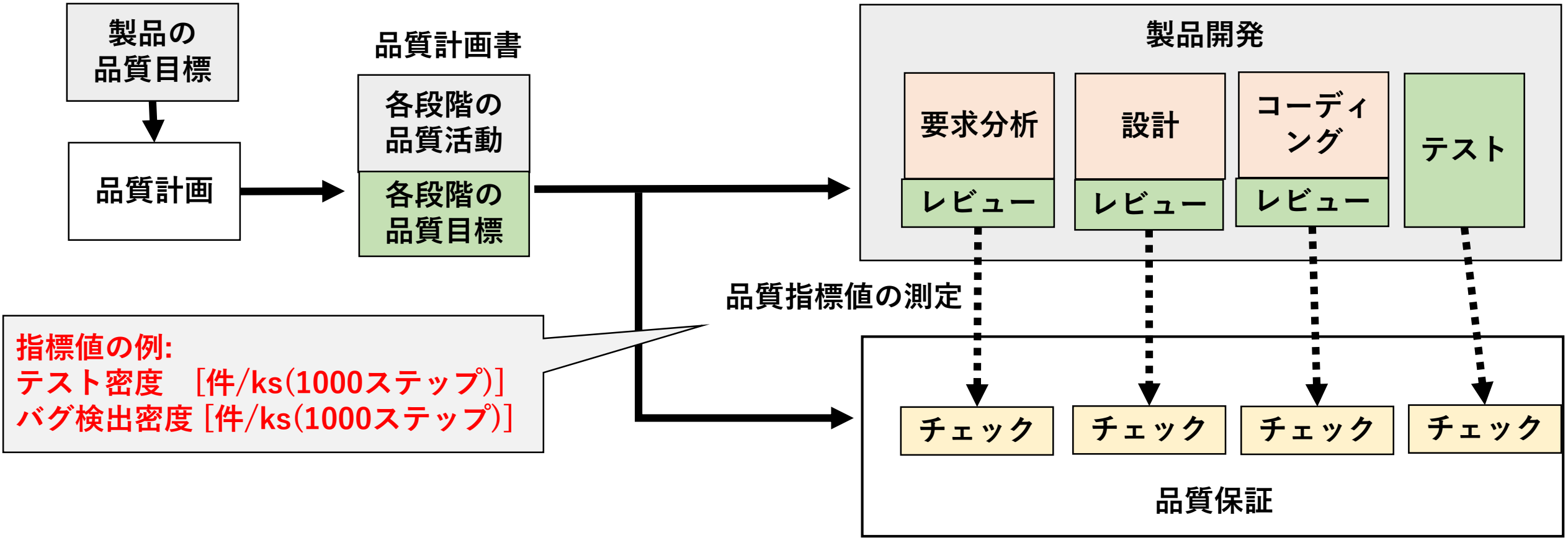
品質特性に基づいた測定可能な指標を製品開発時に定義し、指標の目標値を定めることで、製品のもつべき品質が規定できる。

例)機能完全性→機能実装率、時間効率性→応答時間、スループット

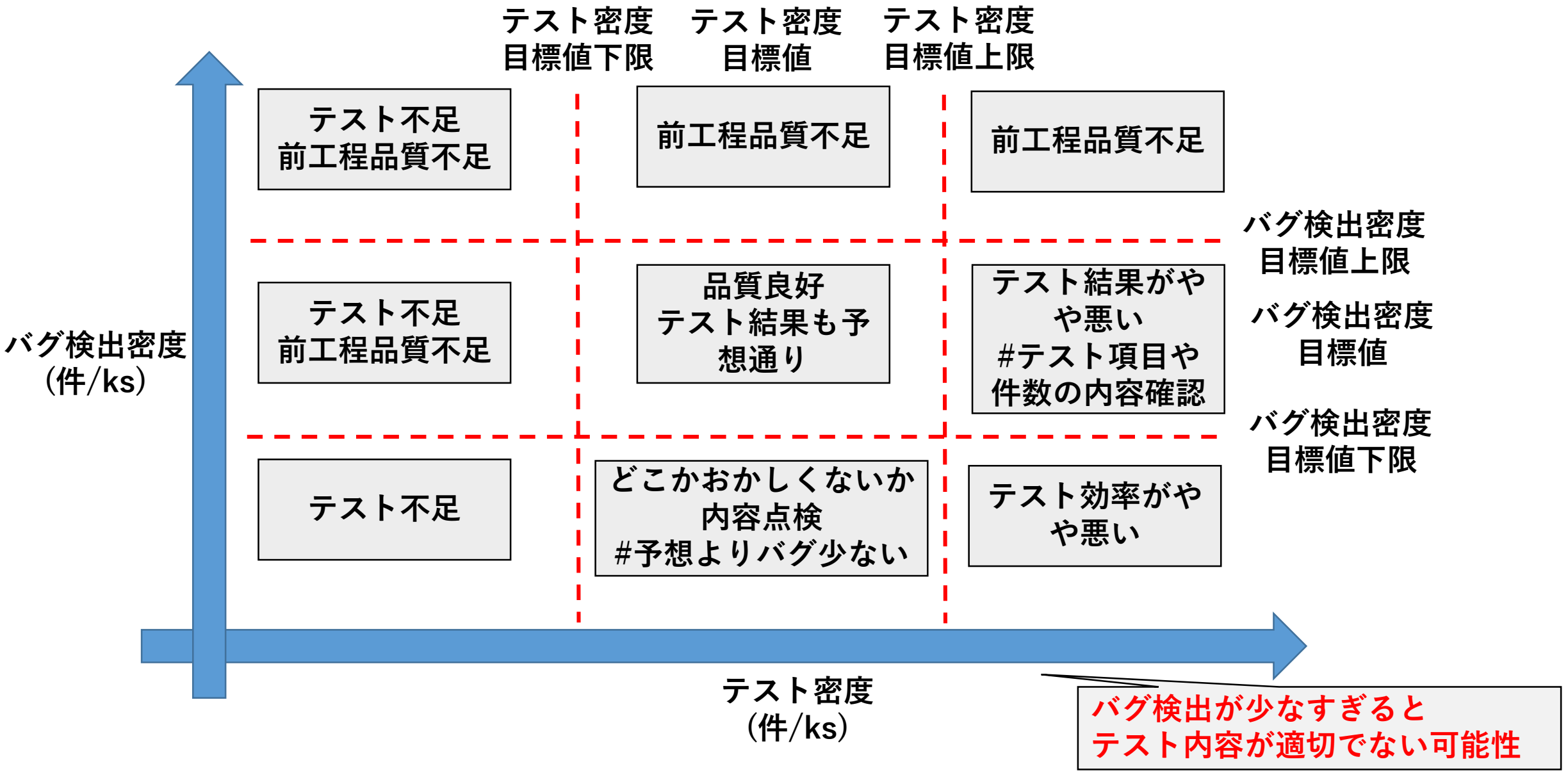
品質保証は、製造する製品が定められた品質目標をクリアしているかどうかを検査する活動となる。

プロダクト品質管理の枠組み

製品の品質は、開発の各段階から作りこむ必要がある。製品の品質目標を達成するために、各開発段階の品質活動と目標を計画し、測定し、制御する活動が必要である。この活動を品質活動と呼ぶ。



テスト密度とバグ検出密度の関係



品質保証の２つのタイプ

品質保証には、開発工程ごとに組み込まれて系統的に行われる組込みタイプと、プログラムを実行させて行うテストタイプがある。

組込みタイプの品質保証:要求分析・設計時からの品質の作りこみを行う目的で、各開発フェーズの中間生産物が一定の品質を持っていることを保証する活動。例)公式レビューにより、各開発レベルの中間生成物である仕様の品質を確認する。

公式レビュー:1)レビュー者の役割と資質、2)レビュープロセスが明確に定義されているレビュー法

テストタイプの品質保証:最終生産物であるプログラムの品質を保証する活動。例)プログラムのテスト実行により、仕様をプログラムが充足するか確認する。

品質保証のプロセス

品質保証のプロセスは、計画と実施から構成される。
a)計画:次の6つの事項を決定する。
①:品質保証組織と役割の定義
②:品質保証作業のカテゴリとレベルの記述
③:品質要求(品質指標)
④:品質保証基準(品質指標に対する目標値、結果を確認する方法)
⑤:検査スケジュール
⑥:検査環境(ハードウェア構成、ソフトウェアツール)
b)実施:準備、実行、評価の3つの作業から構成される。
準備:検査項目の設計と検査環境の準備
実行:実際に製品や半製品を検査し結果を記録する
評価:検査結果が設定した品質保証基準を満たしているか判定する

ソフトウェア構成管理と目的

ソフトウェア開発における生産物は、機能拡張や仕様変更、あるいは不具合処理のため頻繁に変更が行われる。**大規模な開発では、変更を制御しない場合、次のような事態が容易に起こってしまう。**

①プログラム全体のビルド構成が不完全になる。

②プログラム変更がドキュメントに反映されない状態になる。

③同一生産物への複数人での同時編集で、変更が不完全になる。

④実施した検査結果が何を対象にしたものかわからなくなる。

⑤何のために、どの生産物のどこを、どのように変更したのかわからなくなる。

→**ソフトウェア開発における生産物に発生する変更を管理する**
ソフトウェア構成管理

ソフトウェア構成管理(SCM:Software Configuration Management)

ソフトウェア構成管理(SCM)とは、ソフトウェア開発における生産物に発生する変更に対して、一貫性を持つ形態で生産物と生産物間の構成を管理し、変更履歴を追跡可能にするための枠組みである。

以下を行う。

a)管理対象と変更管理手順の管理

　生産物とその変更に対する識別に必要なコード体系を決定

　→バージョン管理

b)ドキュメントとプログラムの変更制御

　開発あるいは保守の対象となる生産物への変更要請を、変更管理手順に従い実際に処理する。

c)ドキュメントとプログラムの構成制御

使用ライブラリのバージョン管理など含む

　ドキュメントとプログラムの対応、不具合修正情報とプログラム/ドキュメントとの対応、プログラムのビルド構成など、構成の一貫性を保持する。

課題13-2:締切7/28

テスト密度とバグ検出密度を算出して、それらの散布グラフを作成せよ。

散布グラフは、目標値の下限上限がわかるようにすること。

散布グラフからテストの状態(例:テスト不足)を分類せよ。

提出:作成した散布グラフ、各モジュールテストの状態

モジュール	単体テスト			結合テスト		
	規模(ks)	テスト項目数(件)	バグ件数(件)	規模(ks)	テスト項目数(件)	バグ件数(件)
P1	4.7	180	28	5.6	64	7
P2	22.2	1332	177	26.6	479	47
P3	13.9	733	97	13.9	220	18
P4	3	178	26	3.6	64	6

モジュール	テスト密度(件/ks)		バグ検出密度(件/ks)	
	下限値	上限値	下限値	上限値
単体テスト	50	70	7	9
結合テスト	15	20	1.5	2

今日の内容

1)プロジェクト管理(WBSなど)の続き

2)品質管理

3)ソフトウェア開発規模と見積り

ソフトウェア開発規模と工数見積り

- ソフトウェア開発の計画段階では、下記1)2)を見積もる。

1)開発対象の規模　ファンクションポイントやソースコードライン数

2)(1で見積もった)規模をもとにした工数　人月(=開発人数×開発月数)

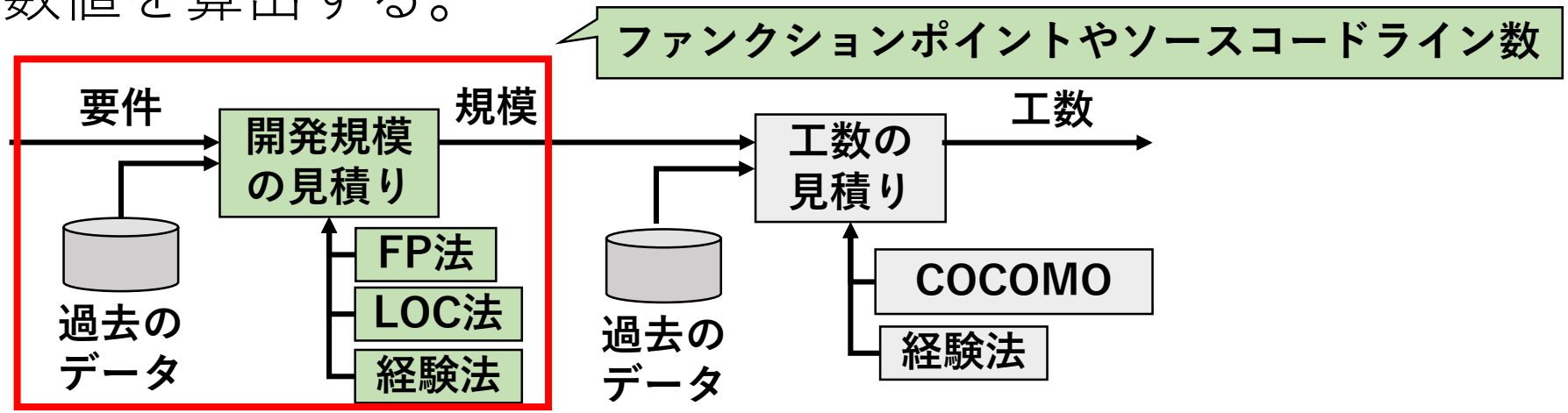
規模と工数をもとに、開発コストの算出、開発スケジュールの設定、開発要員の確保が行われる。

- ソフトウェア開発規模と工数は、設計が進んだ段階においても見積もられ、開発工数の進捗管理に活用される。さらに、規模と工数は開発完了時に算出され、次の開発プロジェクトへの改善に活用される。

進捗が予定通りかの確認

ソフトウェア開発における見積り

- 開発規模は、ソフトウェアの要件と過去のデータをもとに、各種技法を使って見積り、この値を工数に変換して経験や過去の実績データと比較して妥当な工数値を算出する。



- ソフトウェア開発規模は、ソフトウェアの特性、使用する言語の種類などに依存し、開発する人間的要員にも影響を受けやすい。そのため、見積りには一定の尺度がなく、経験が必要となる場合が多い。
- 主なソフトウェア規模の見積り方法として以下がある。

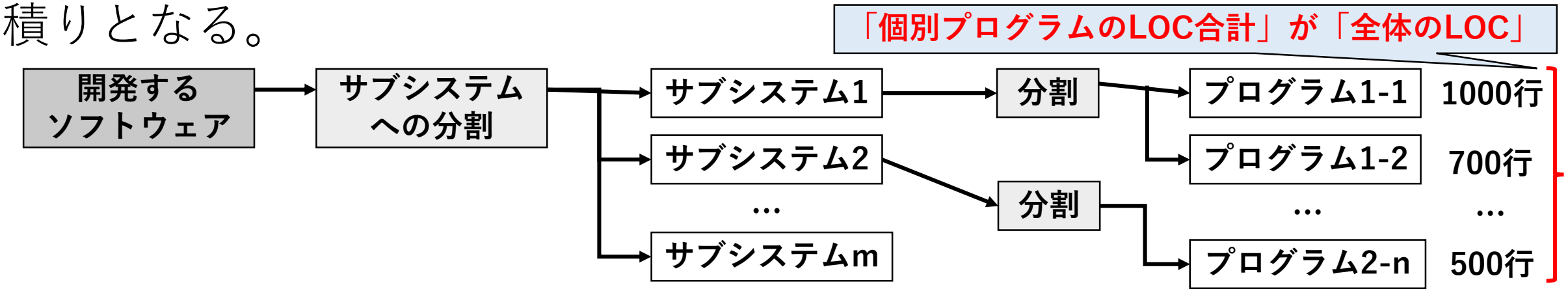
1)LOC法:ソースコードの行数を尺度

2)ファンクションポイント法(FP法):ソフトウェアの持つ機能により規模の大小を表す方法

LOC法:Line Of Code

LOC法:ソフトウェアのソースコードの行数を尺度として、ソフトウェア規模を直接算出する方法である。直感的に規模を把握できることから、実際によく使用されている。

- 過去の開発経験や蓄積データ、類似のソフトウェアと比較し、その要求機能と想定される実装構造の差異により類推してLOCを算出する。開発には様々なプログラミング言語が使用され、1行の重みが異なるが、LOC法では1行は1行として規模尺度としている。
- ソフトウェア開発プロジェクトでは通常、システムは複数のサブシステムに、さらにサブシステムは複数のプログラムに分解され開発される。つまり、全プログラムのLOCの和が開発プロジェクトのソフトウェアのLOC見積りとなる。



ファンクションポイント法

ファンクションポイント法(FP法):開発するソフトウェアの規模をその機能数(ファンクションポイント)で計測する方法であり、機能を画面入出力、**帳票出力**、ファイルアクセス、外部インタフェースとのやり取りなどの数で測る。

帳票:伝票や帳簿(例:部署の固定資産一覧)

FP法では、入出力の数や処理するファイルの数をカウントすることによりソフトウェア規模をFP値で表す間接的な計測方法で、**ユーザ側から見た機能によって規模を見積もる**。

- FP法には以下の特徴がある。

①ユーザの視点に立った機能量により見積もるため、開発手法やプログラミング言語に依存しない。

②機能要件と規模見積り結果との対応が理解しやすいため、開発初期段階から適用できる。

③事務処理用ソフトウェアの機能には、画面入出力、帳票出力が多く含まれ、画面、帳票の中のデータ項目はデータベースへのアクセスによるものが多いため、FP法の適用性が高い。

ファンクションポイント法による見積り

FP法の見積りでは、ソフトウェア機能を5つの機能タイプに分類し、機能タイプごとに複雑度を判定する。

[機能タイプ]

ユーザが使用する入力画面

①**外部入力:**「画面や他システムからの入力データの処理」と「内部論理ファイルの更新機能」

ユーザが使用する出力画面・帳票

②**外部出力:**内部論理ファイルを処理して画面や外部へ出力する機能

③**外部照会:**内部論理ファイルの更新を伴わない参照のみの機能

一覧表示

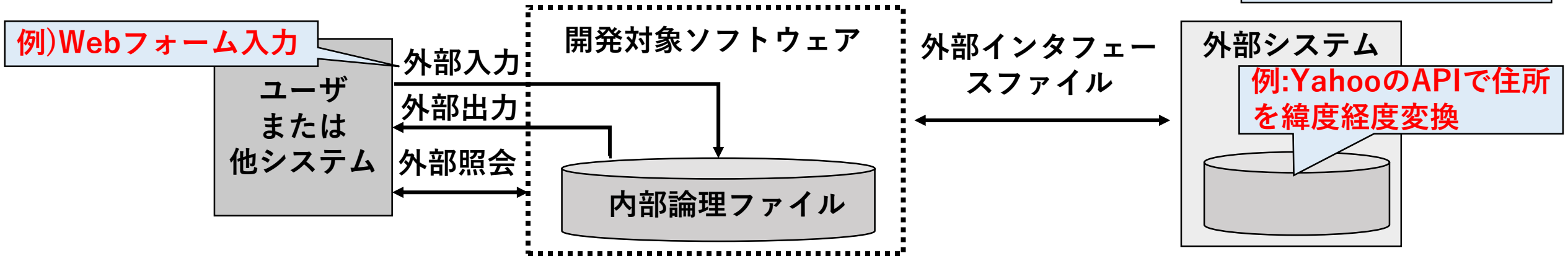
④**内部論理ファイル:**開発対象ソフトが持っているファイルで、開発対象ソフトが保守管理の責任を持つ

ファイルやデータベース

#例:商品マスタ、顧客マスタ、在庫マスタなど

⑤**外部インタフェースファイル:**外部のファイルを参照する機能

他システムとの連携
例:Web API



ファンクションポイント法による見積り1

FP法では、以下の手順で見積りを行う。

①計測する境界の設定:計測対象のソフトウェア範囲と外部との境界を設定

②機能タイプごとのユーザ機能数のカウントと複雑度判定:5つの機能タイプごとに複雑度を判定する。各ユーザ機能ごとに複雑度を判定する。複雑度は、ユーザ機能が参照したファイル種別とユーザ機能が処理するデータ項目数の交点から求める。

例)アクセスする
データベースの表数

例)アクセスするカラム数

・ファイルやデータベース
・他システムとの連携

各表のデータ数などを考慮
した見積もりは出来ません

例)アクセスする
データベースの表数
#顧客表→1

例)アクセスするカラム数
顧客ID,顧客名→2

内部論理ファイル
外部インタフェースファイル

参照したファイル種別	データ項目数		
	1～4	5～15	16～
<2	単純	単純	平均
2	単純	平均	複雑
>2	平均	複雑	複雑

参照したファイル種別:顧客表
データ項目数:顧客ID,顧客名
→複雑度は単純

ファンクションポイント法による見積り2

③未調整FPの計算:5つの機能タイプのカウント数を複雑度に対応した係数を乗じて、5つの機能タイプごとのFPを算出し、その合計である未調整FPを算出する。

未調整FP=C1+C2+C3+C4+C5

機能タイプ	複雑度			計
	単純	平均	複雑	
外部入力	□×3 ＋	□×4 ＋	□×6	=C1
外部出力	□×4 ＋	□×5 ＋	□×7	=C2
内部ファイル	□×7 ＋	□×10 ＋	□×15	=C3
外部インタフェースファイル	□×5 ＋	□×7 ＋	□×10	=C4
外部照会	□×3 ＋	□×4 ＋	□×6	=C5

各機能タイプの複雑度を乗じて算出

内部論理ファイル
外部インタフェースファイル

参照したファイル種別	データ項目数		
	1～4	5～15	16～
<2	単純	単純	平均
2	単純	平均	複雑
>2	平均	複雑	複雑

外部入力、外部出力などの換算表は教科書に記載があります

ファンクションポイント法による見積り3

FP法では、以下の手順で見積りを行う。

④システム特性係数の計算:以下の式で計算する。

システム特性係数=0.65+(0.01×T)

T(最大70)=T1+T2+T3+…+T14

Tは、開発対象のソフトが動作するシステムの環境がソフト機能に与える影響度を示す。
システム特性係数は、最小値0.65と最大値1.35の間となる。

システム特性	影響度 0～5	システム特性	影響度 0～5
オンライン処理の度合	T1	マスタファイルのオンライン	T8
分散処理の度合	T2	更新処理の複雑度合	T9
性能重要度合	T3	プログラムの再利用性の考慮度合	T10
ハードウェア制約の大小	T4	導入の容易性の度合	T11
トランザクション発生率	T5	運用の容易性	T12
オンラインデータ入力使用度	T6	複数拠点への導入容易性	T13
エンドユーザ作業効率考慮の度合	T7	変更の容易性	T14

⑤FPの算出:求めるFPは、(未調整FP)×(システム特性係数)となる。

ファンクションポイント法の課題

FP法には、以下のような課題がある

①制御・通信・科学計算ソフトウェアなどは、前述の5つの機能に分類できないため適用が難しい

②システム特性係数算出における影響度判定には主観的要素が入り、また複雑度の判定には経験を要するため、計測する人によって値が異なる可能性がある。

③計測するソフトウェアの範囲と外部との境界設定が適切でない場合やサブシステムの分け方が妥当でない場合には、二重にカウントしてしまう部分が生じる。

ファンクションポイントとLOCとの変換

FPとソースコード量との比率が、各種言語について実測データの分析により求められている。主な言語に対して、1ファンクションポイントあたりのLOCを示したものが以下である。Cの1LOCはアセンブラ言語の約2.5倍の機能記述となることを表している。

FP法とLOC法の両方で見積もった場合の比較検討の参考となる。

プログラミング言語	1FPあたりのLOC(最頻値)
アセンブラ言語	320
C	128
COBOL	105
Pascal	91
オブジェクト指向言語	30

工数見積り

開発するソフトウェアの規模は、ソフトウェア仕様に基づくLOC法やFP法などによって計測される。開発工数は、規模及び開発プロジェクトの内容に影響される。

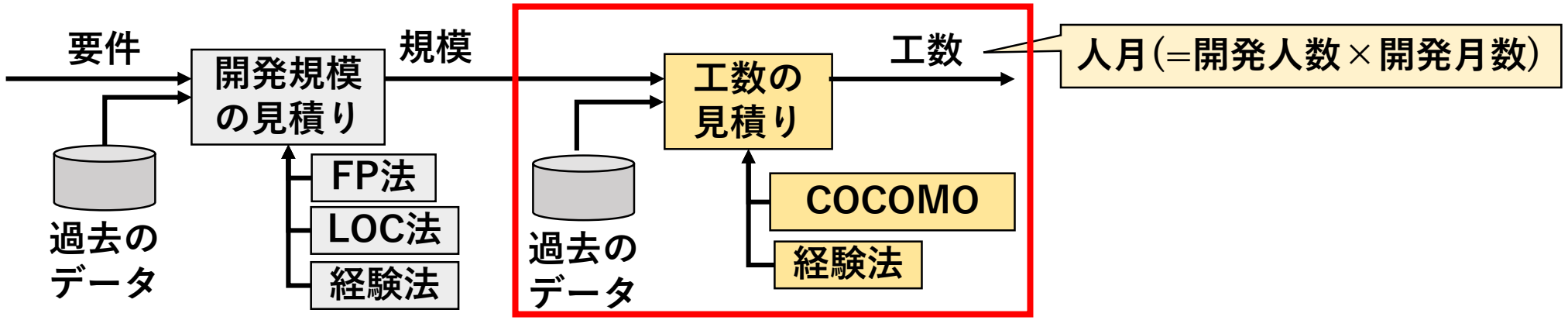
工数見積りには、以下のような技法がある。

a)類推法:過去に行った類似のプロジェクトの実績データをもとに類推し、開発対象プロジェクトの工数を見積もる方法

b)積上げ法:開発プロジェクトで必要とされる1つ1つの作業を取り出し、各作業単位ごとに工数を見積り、それらを合計して全体の工数とする。外部設計で、入出力画面や帳票の種類がほぼ決まれば、作業の単位とできる。

c)標準タスク法:開発に必要な１つ１つの作業をあらかじめ標準タスクとして設定し、標準タスクごとに作業の規模や複雑度を考慮して工数を決めておく。開発プロジェクトで必要とされる作業を標準タスクの集まりに分類し、集計により工数を見積もる。

d)COCOMO:LOCで表されるソフトウェア規模から開発工数と開発期間を算出する方法である。



COCOMO(1/2)

1981年に提案された工数見積りモデルで、数十個の開発プロジェクトの実績データをもとにしている。COCOMOでは、開発するソフトウェアの規模をKLOC(kilo LOC)で表し、モデル式から開発工数と開発期間を算出する。

E:開発工数[人月], D:最適開発期間[月], a,b,c,d:係数, f:影響要因

$E = a \times KLOC^{b} \times f$

$D = c \times E^{d}$

COCOMOでは、開発のどの時点で、どの単位で見積りを行うかによってモデルを３種にわけ、さらに各モデルを開発形態の違いにより３つの開発モードに分けて、モデル式の係数を設定している。

COCOMO(2/2)

a)見積り単位による分類

①基本COCOMO:主としてLOCだけから工数を算出する。開発段階の初期で使用する。

②中間COCOMO:要求仕様が決まり、全体をいくつかのコンポーネントに分割できる段階でコンポーネントごとに工数を見積り、合計を算出する。

③詳細COCOMO:設計が進み、開発ソフトウェアの構造が確定した時点での見積りに使用する。開発ソフトウェアをシステム、サブシステム、モジュールの3段階の単位に分け、影響要因を乗じて見積る。

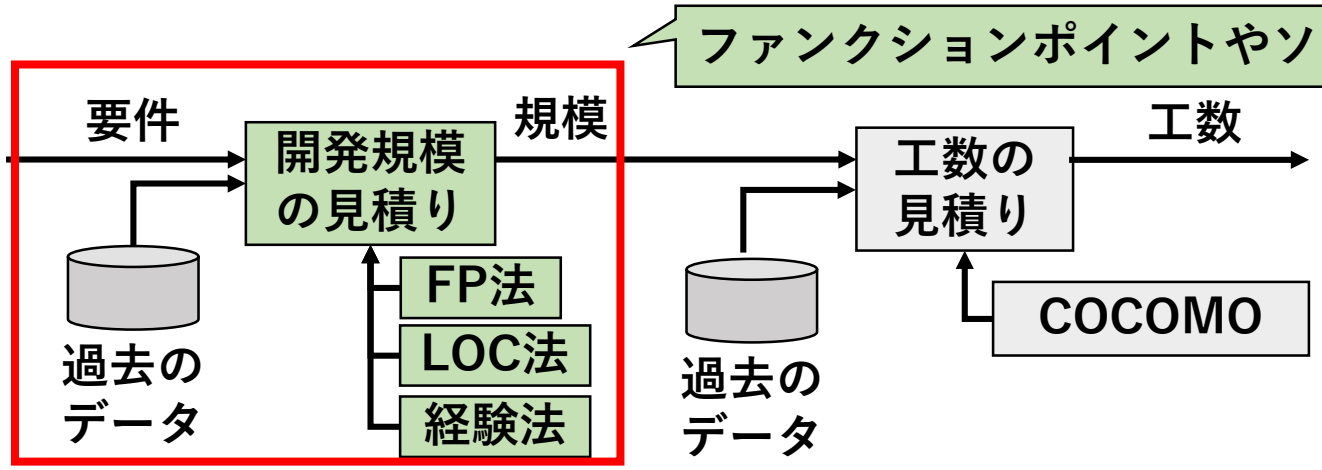
b)開発モードによる分類

①組織モード:レベルの高い少人数の開発チームで特定のソフトウェアを開発する場合の開発モード

②組込みモード:稼働条件の厳しい大規模なリアルタイムシステムのようなソフトウェア開発を、大人数の開発プロジェクトで行う場合の開発モード

③半組込みモード:組織モードと組込みモードの中間的な形態の開発モード。一般の事務処理システムなど、多くの開発形態が該当する。

COCOMOによる工数見積り例



プログラミング言語	1FPあたりのLOC(最頻値)
アセンブラ言語	320
C	128

FP法なら規模をLOCへ変換

E:開発工数[人月], D:最適開発期間[月], a,b,c,d:係数, f:影響要因

$E = a \times KLOC^{b} \times f$

ソフトウェアの規模・性質:例)a:3.2,b:1.05,f:0.38

$D = c \times E^{d}$

ソフトウェアの規模・性質:例)0.35

ソフトウェアの規模・性質:例)2.5

開発工数(人月)が1000の場合の見積り

$D = 2.5 \times 1000^{0.35} = 28.0504 \fallingdotseq 28$か月

課題13-3:締切7/28

以下のシステム開発のFPを見積もり、C言語で開発した場合の開発工数と最適開発期間を算出せよ。なお、システム特性係数は合計40とし、式の係数は前スライドの値をもちいること。**提出**:計算結果と計算の途中式

外部入力:単純×2

外部出力:単純×1

外部照会:単純×1

内部ファイル:単純×1

外部インタフェースファイル:単純×1

機能タイプ	複雑度			計
	単純	平均	複雑	
外部入力	□×3	+ □×4	+ □×6	=C1
外部出力	□×4	+ □×5	+ □×7	=C2
内部ファイル	□×7	+□×10	+ □×15	=C3
外部インタフェースファイル	□×5	+ □×7	+ □×10	=C4
外部照会	□×3	+ □×4	+ □×6	=C5

FPを求めて、開発工数算出し、最適開発期間を求めてください