Software Engineering

Part 1 Software Process

Chapter 4 Process Models

Reproduced by Ning Li , 2022

Contents

4.1 Prescriptive process models

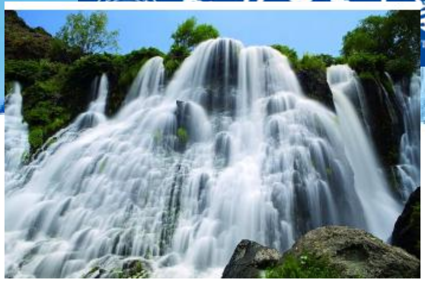
- Prescriptive process models advocate an orderly approach to software engineering

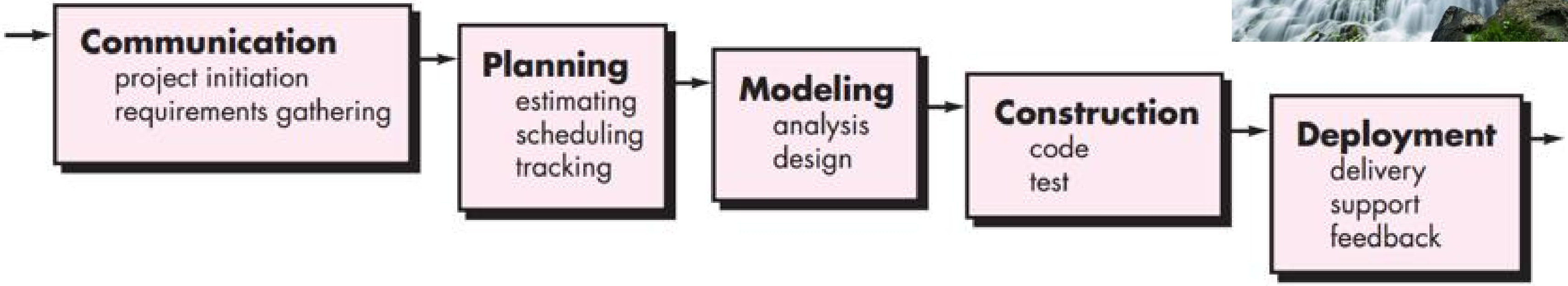
That leads to a few questions …

- If prescriptive process models strive for structure and order, are they inappropriate for a software world that thrives on change?

- Yet, if we reject traditional process models (and the order they imply) and replace them with something less structured, do we make it impossible to achieve coordination and coherence in software work?

4.1.1 The Waterfall Model





Linear flow: Waterfall model (classic life cycle)

Focus: sequential

Applicable case:

- Requirements are fixed / not changing frequently
- Requirement is clear
- Application is not complicated and big
- Environment is stable
- Technology and tools used are stable
- Resources are available and trained

主观题 10分

设置

What are the disadvantages of this waterfall model?

正常使用主观题需2.0以上版本雨课堂

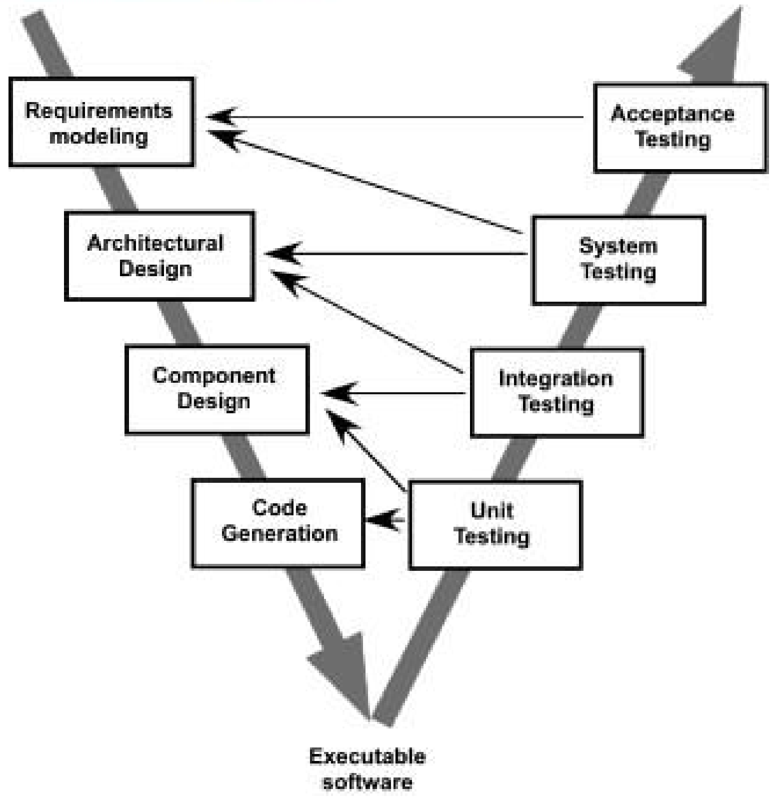
作答

4.1.1 The Waterfall Model

What are the disadvantages of this model?

1. Real projects rarely follow the sequential flow that the model proposes. It’s not easy to do iteration.
2. It is often difficult for the customer to state all requirements explicitly.
3. The customer must have patience. A working version of the program(s) will not be available until late in the project time span.

4.1.1 The V-Model (Waterfall variation)

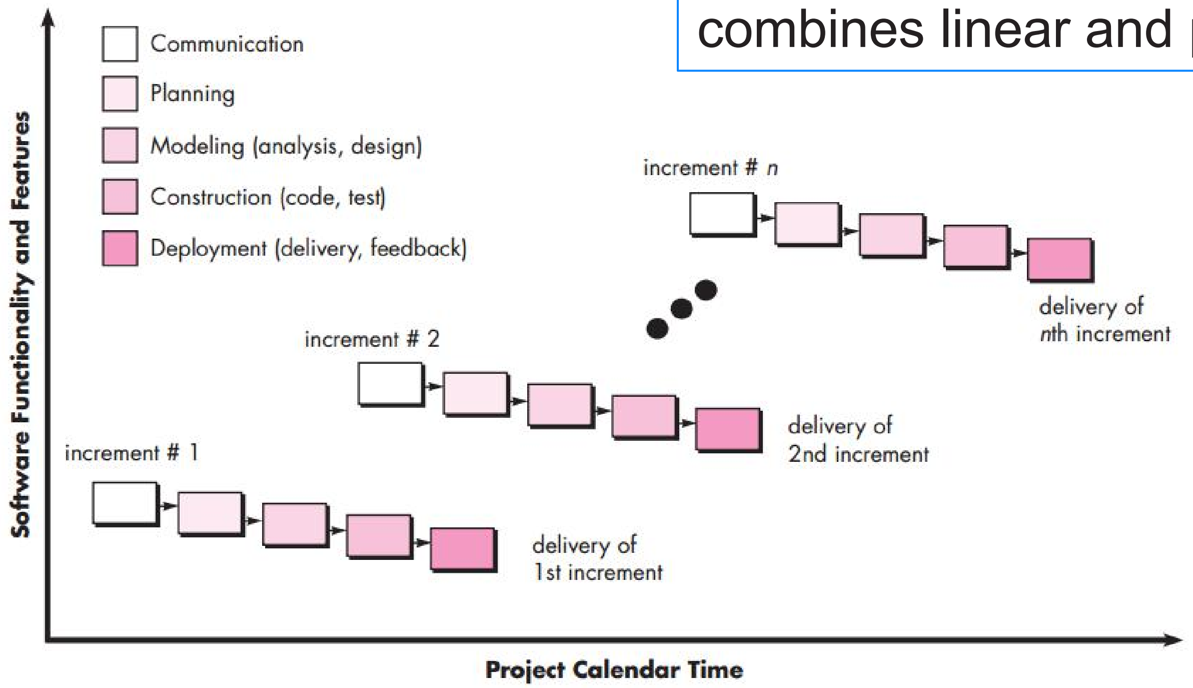


Problems:

1. Real projects rarely follow the sequential flow that the model proposes.
2. It is often difficult for the customer to state all requirements explicitly.
3. The customer must have patience.

4.1.2 The Incremental Model

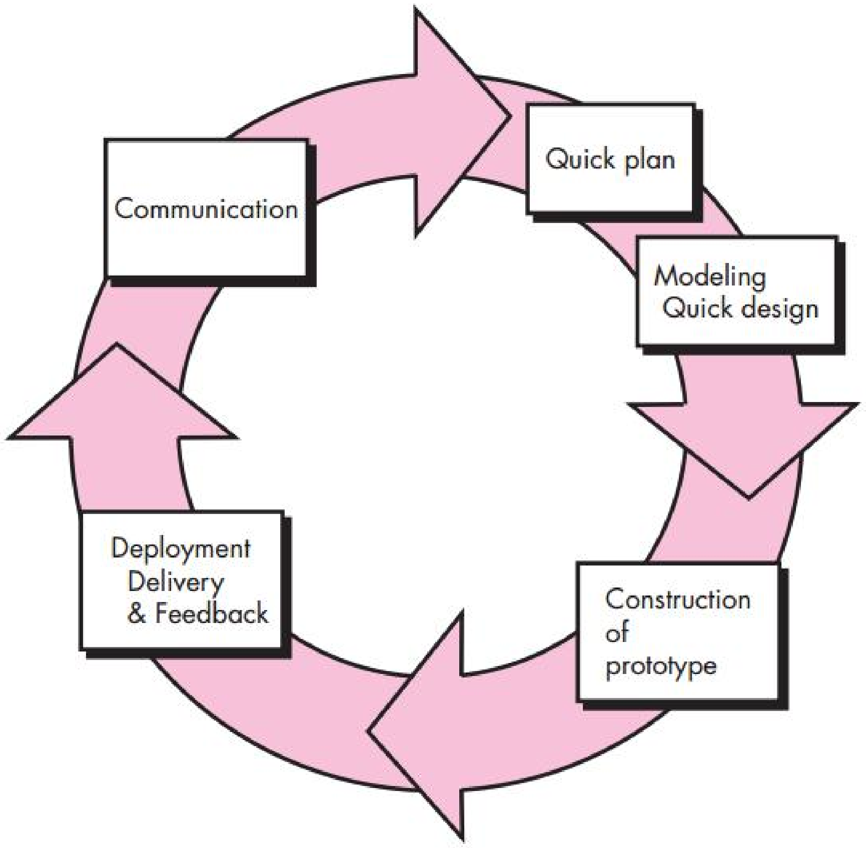
combines linear and parallel process


Communication
Planning
Modeling (analysis, design)
Construction (code, test)
Deployment (delivery, feedback)
increment # n
delivery of nth increment
increment # 2
delivery of 2nd increment
increment # 1
delivery of 1st increment
Software Functionality and Features
Project Calendar Time

Example - word-processing software:

1st increment(core):	basic file management, editing and doc production.
2nd increment:	sophisticated editing and doc production
3rd increment:	spelling and grammar checking
4th increment:	advanced page layout capability

4.1.4 Evolutionary Models: Prototyping



Applicable case:

1. Customer defines a set of general objectives for software, but does not identify detailed requirements for functions and features.

2. Developer may be unsure of the efficiency of an algorithm, ability of an operating system, or the form that human-machine interaction should take.

Key: It is a mechanism for identifying software requirements.

投票 最多可选1项

设置

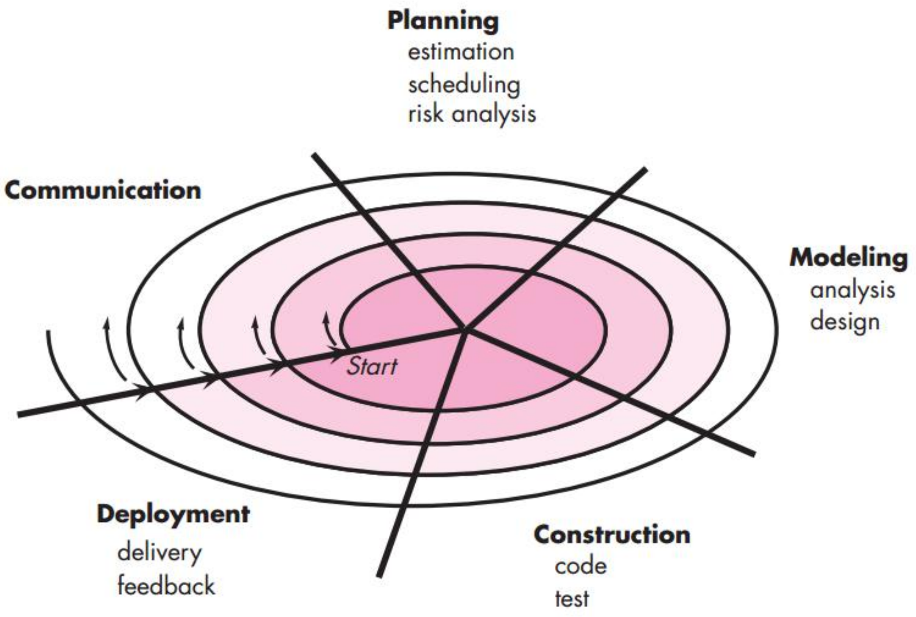
How to deal with the created prototype?

A throwaways

B evolutionary

提交

4.1.4 Evolutionary Models: The Spiral



Circuit around spiral:

Circuit1:
  Product specification.

Circuit2:
  Prototype

Circuit3:
  More sophisticated versions of software.

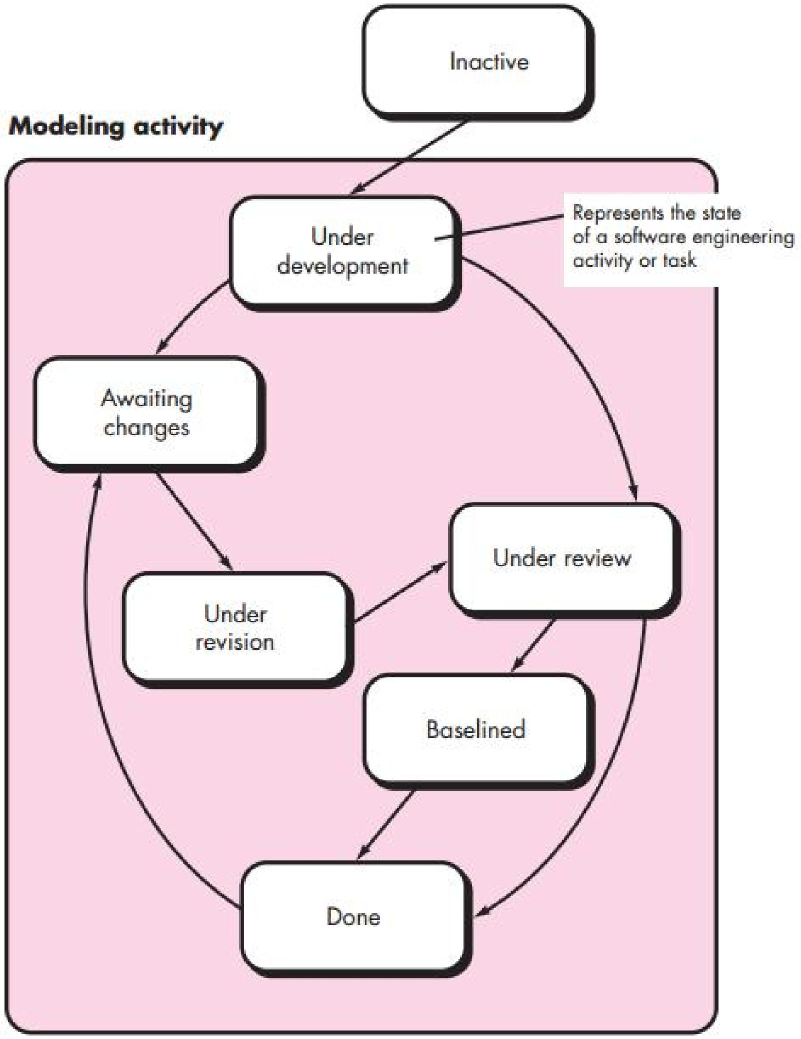
Circuit4:
  …

(Each pass starts with plan)

Key: Risk analysis (iterative prototype waterfall , risk-driven model)

the project manager adjusts the planned number of iterations required to complete the software

4.1.4 Concurrent Models



- Applicable to all types of software development.

Example:
- communication activity
- modeling activity

- Provides an accurate picture of the current state of a project.

Emphasize：
- flexibility,
- extensibility
- speed of development

4.1 Process Models

- Waterfall Model
  - Basic waterfall model
  - V-Model (Waterfall variation)
- Incremental Model
- Evolutionary Models
  - Prototyping
  - Spiral
- Concurrent

投票 最多可选1项

设置

Which model is considered to be the best approach to software development in a modern context?

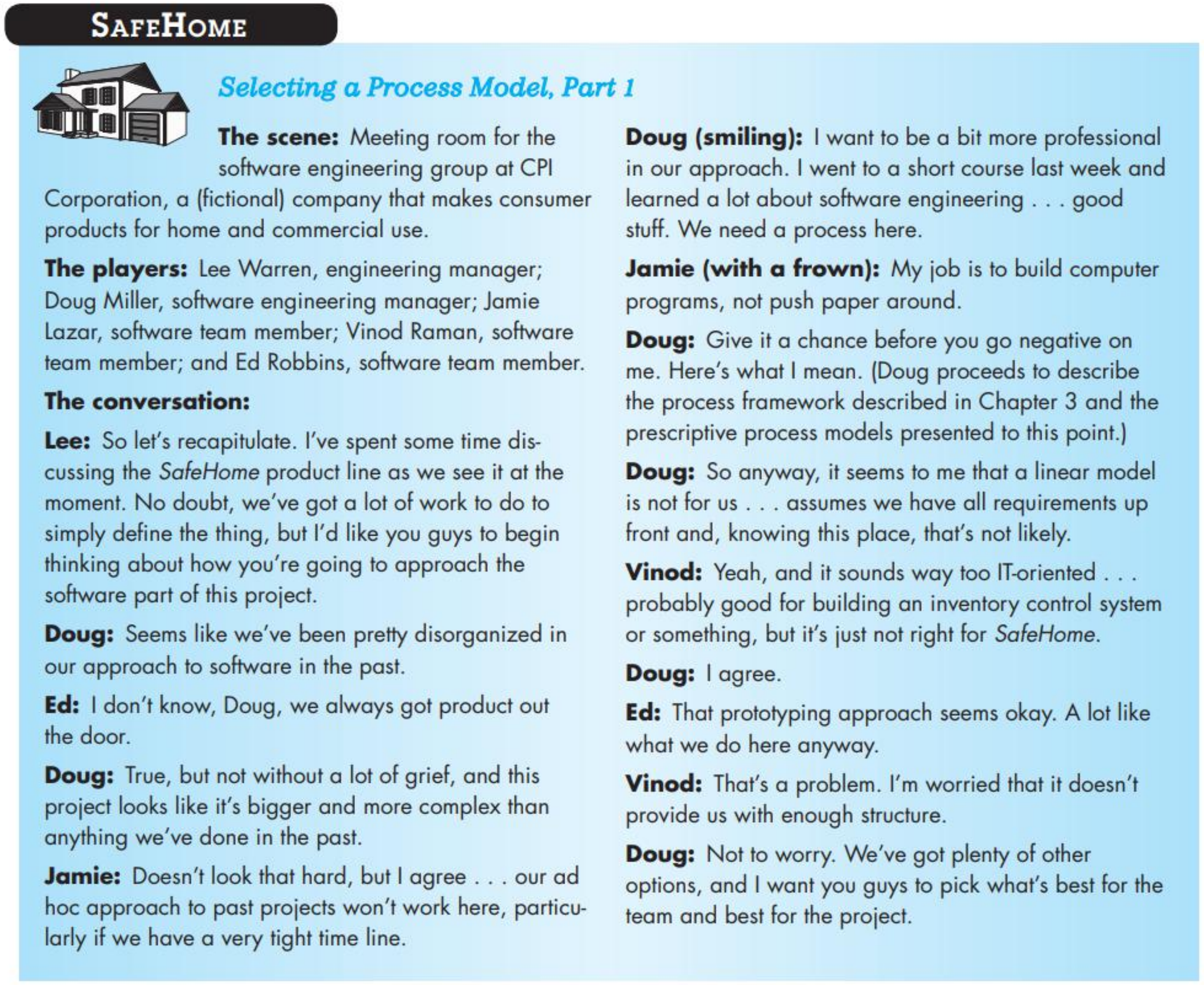
A Waterfall model

B Evolutionary model

提交

4.1 Process Models

Please read SafeHome example:
Selecting a Process Model, Part 1 (P47)



SAFEHOME

Selecting a Process Model, Part 1

The scene: Meeting room for the software engineering group at CPI Corporation, a (fictional) company that makes consumer products for home and commercial use.

The players: Lee Warren, engineering manager; Doug Miller, software engineering manager; Jamie Lazar, software team member; Vinod Raman, software team member; and Ed Robbins, software team member.

The conversation:

Lee: So let's recapitulate. I've spent some time discussing the *SafeHome* product line as we see it at the moment. No doubt, we've got a lot of work to do to simply define the thing, but I'd like you guys to begin thinking about how you're going to approach the software part of this project.

Doug: Seems like we've been pretty disorganized in our approach to software in the past.

Ed: I don't know, Doug, we always got product out the door.

Doug: True, but not without a lot of grief, and this project looks like it's bigger and more complex than anything we've done in the past.

Jamie: Doesn't look that hard, but I agree . . . our ad hoc approach to past projects won't work here, particularly if we have a very tight time line.

Doug (smiling): I want to be a bit more professional in our approach. I went to a short course last week and learned a lot about software engineering . . . good stuff. We need a process here.

Jamie (with a frown): My job is to build computer programs, not push paper around.

Doug: Give it a chance before you go negative on me. Here's what I mean. (Doug proceeds to describe the process framework described in Chapter 3 and the prescriptive process models presented to this point.)

Doug: So anyway, it seems to me that a linear model is not for us . . . assumes we have all requirements up front and, knowing this place, that's not likely.

Vinod: Yeah, and it sounds way too IT-oriented . . . probably good for building an inventory control system or something, but it's just not right for *SafeHome*.

Doug: I agree.

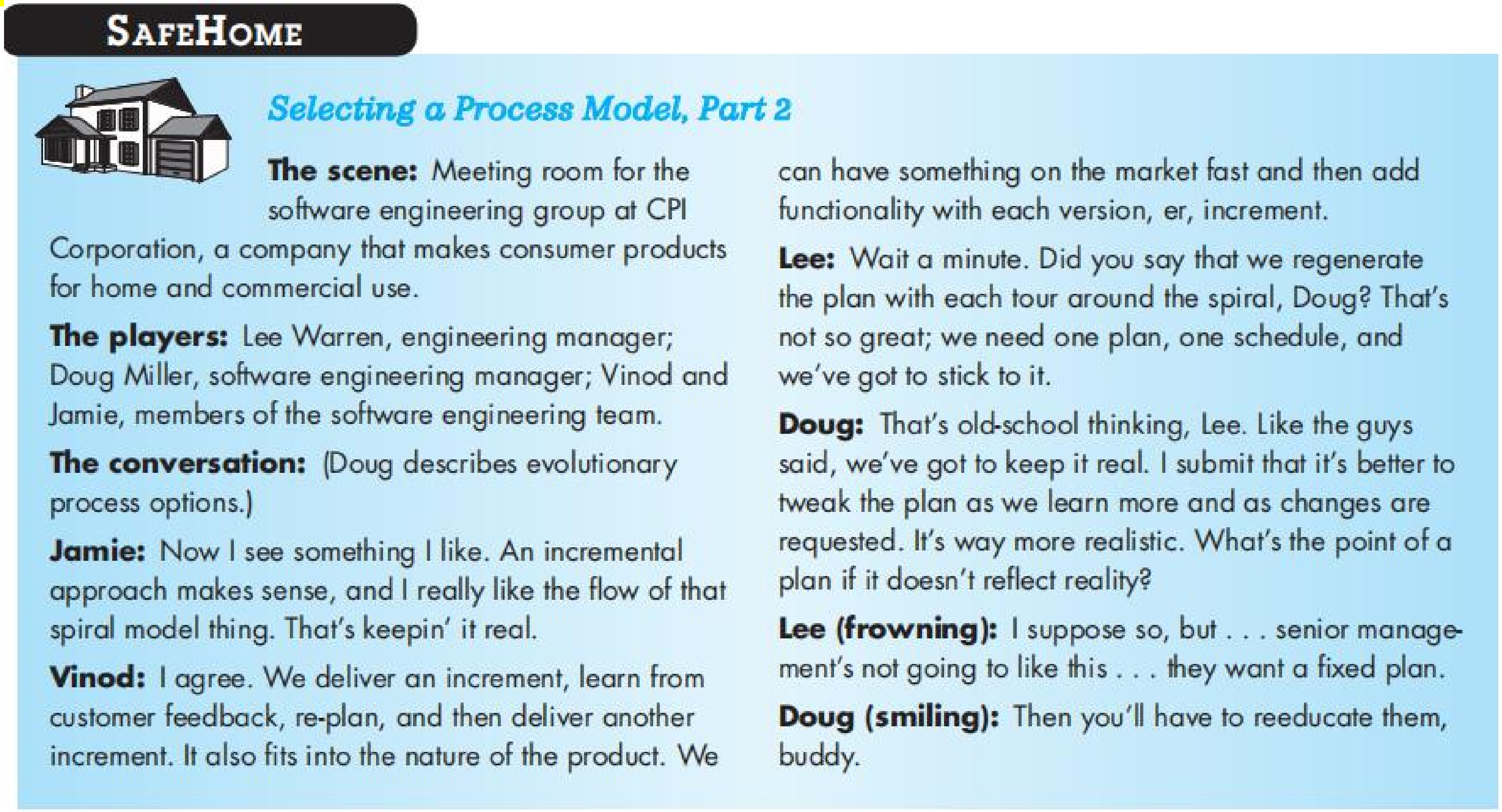
Ed: That prototyping approach seems okay. A lot like what we do here anyway.

Vinod: That's a problem. I'm worried that it doesn't provide us with enough structure.

Doug: Not to worry. We've got plenty of other options, and I want you guys to pick what's best for the team and best for the project.

4.1 Process Models

Please read SafeHome example:
Selecting a Process Model, Part 2 (P48)



SafeHome

Selecting a Process Model, Part 2

The scene: Meeting room for the software engineering group at CPI Corporation, a company that makes consumer products for home and commercial use.

The players: Lee Warren, engineering manager; Doug Miller, software engineering manager; Vinod and Jamie, members of the software engineering team.

The conversation: (Doug describes evolutionary process options.)

Jamie: Now I see something I like. An incremental approach makes sense, and I really like the flow of that spiral model thing. That's keepin' it real.

Vinod: I agree. We deliver an increment, learn from customer feedback, re-plan, and then deliver another increment. It also fits into the nature of the product. We can have something on the market fast and then add functionality with each version, er, increment.

Lee: Wait a minute. Did you say that we regenerate the plan with each tour around the spiral, Doug? That's not so great; we need one plan, one schedule, and we've got to stick to it.

Doug: That's old-school thinking, Lee. Like the guys said, we've got to keep it real. I submit that it's better to tweak the plan as we learn more and as changes are requested. It's way more realistic. What's the point of a plan if it doesn't reflect reality?

Lee (frowning): I suppose so, but . . . senior management's not going to like this . . . they want a fixed plan.

Doug (smiling): Then you'll have to reeducate them, buddy.

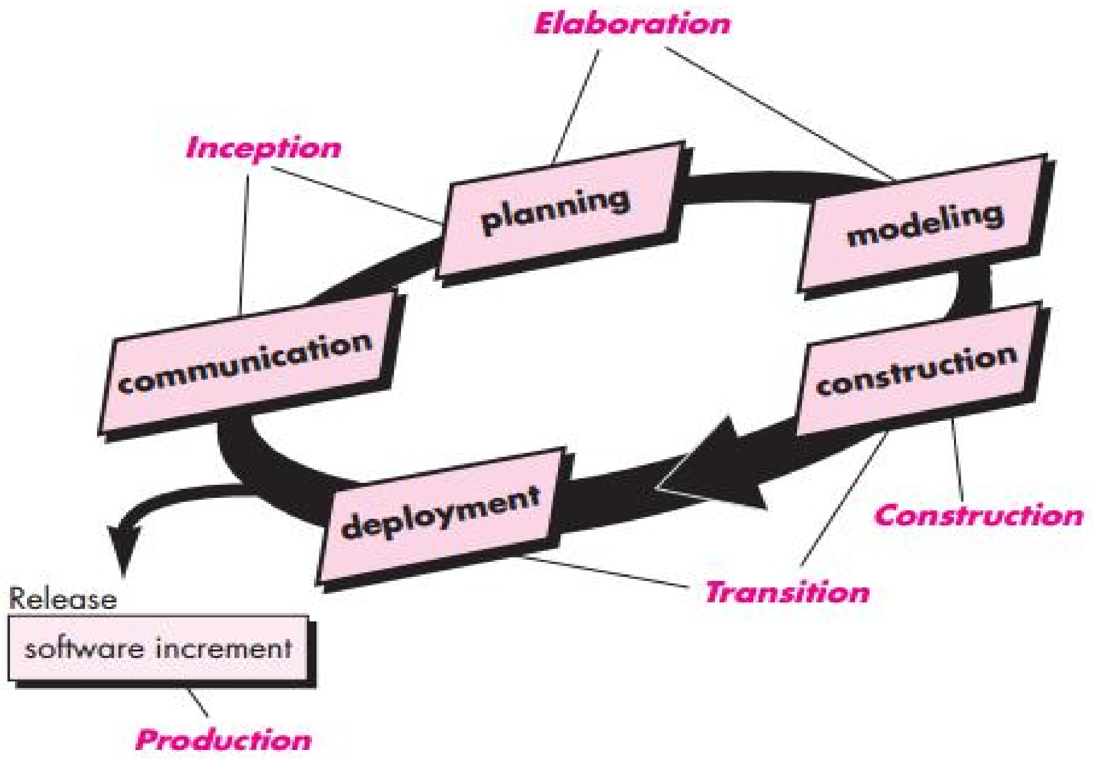
4.1 Process Models

What is the difference between Incremental Model and Spiral model?

4.2 Specialized Process Models

- Component based development—the process to apply when reuse is a development objective
- Formal methods—emphasizes the mathematical specification of requirements
- AOSD—provides a process and methodological approach for defining, specifying, designing, and constructing *aspects*

4.3 The Unified Process (UP)



By 1997, UML became a industry standard for OO software development.

4.3 The Unified Process (UP)

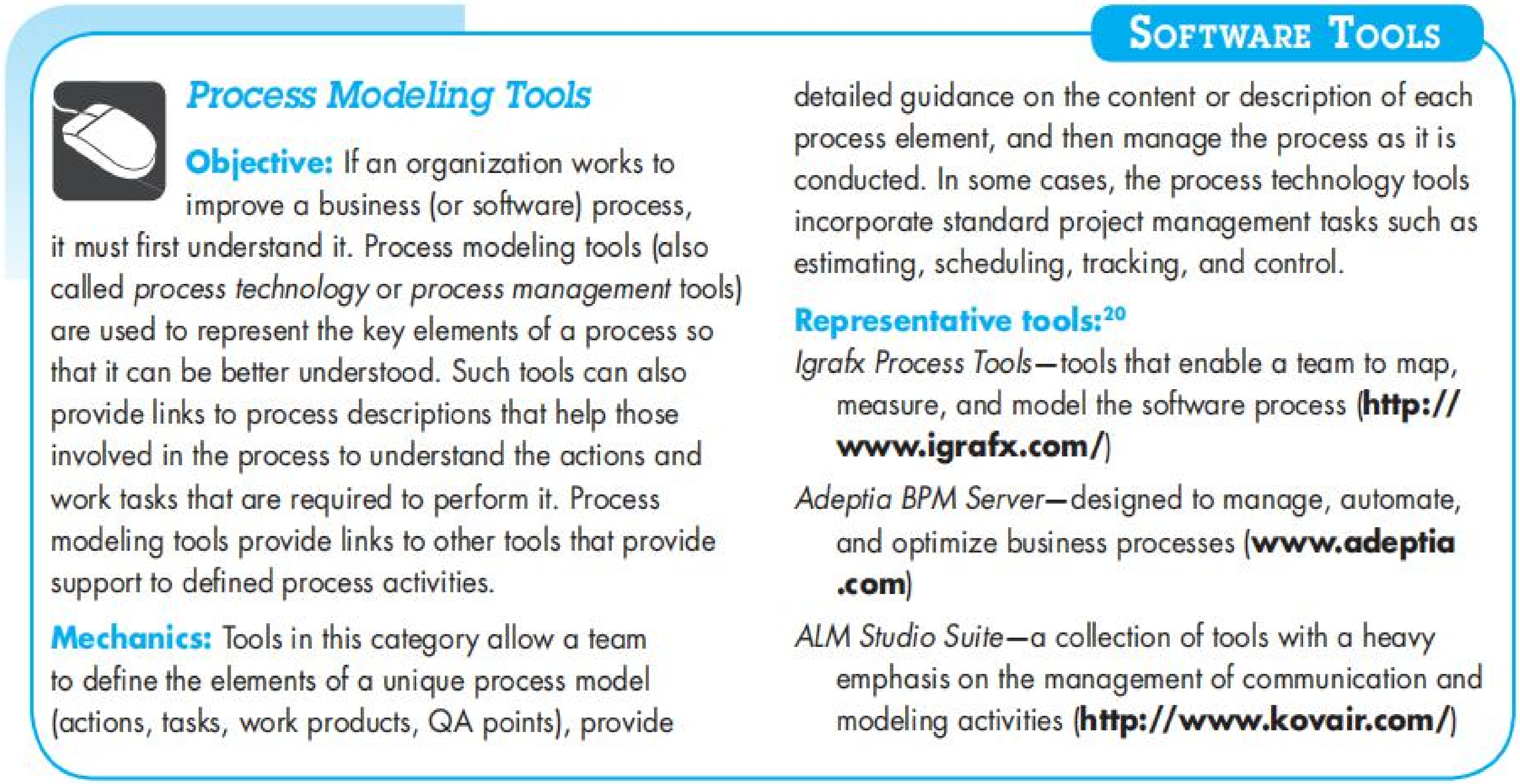
UP phase	Task
Inception	customer communication and planning activities
Elaboration	communication and modeling activities
Construction	construction activity
Transition	the latter stages of the generic construction activity the first part of the generic deployment (delivery and feedback) creates the necessary support information (help etc.)
Production	deployment activity (monitor and support) defect reports and requests for changes are submitted and evaluated

4.4 PERSONAL AND TEAM PROCESS MODELS

- *Personal Software Process* (PSP)
  - Planning
  - High-level design
  - High-level design review
  - Development
  - Postmortem

- Team Software Process (TSP)
  - self-directed
    (organizes itself to produce high-quality software)

4.5 PROCESS TECHNOLOGY (tools)



SOFTWARE TOOLS

Process Modeling Tools

Objective: If an organization works to improve a business (or software) process, it must first understand it. Process modeling tools (also called *process technology* or *process management* tools) are used to represent the key elements of a process so that it can be better understood. Such tools can also provide links to process descriptions that help those involved in the process to understand the actions and work tasks that are required to perform it. Process modeling tools provide links to other tools that provide support to defined process activities.

Mechanics: Tools in this category allow a team to define the elements of a unique process model (actions, tasks, work products, QA points), provide detailed guidance on the content or description of each process element, and then manage the process as it is conducted. In some cases, the process technology tools incorporate standard project management tasks such as estimating, scheduling, tracking, and control.

Representative tools:[20]

Igrafx Process Tools—tools that enable a team to map, measure, and model the software process (**http://www.igrafx.com/**)

Adeptia BPM Server—designed to manage, automate, and optimize business processes (**www.adeptia.com**)

ALM Studio Suite—a collection of tools with a heavy emphasis on the management of communication and modeling activities (**http://www.kovair.com/**)

4.6 Product and Process

- If the process is weak, the end product will undoubtedly suffer.

- As creative software professional, you should also derive as much satisfaction from the process as the end product.

You can never derive or understand the full artifact, its context, use, meaning, and worth if you view it as only a process or only a product.

THE END