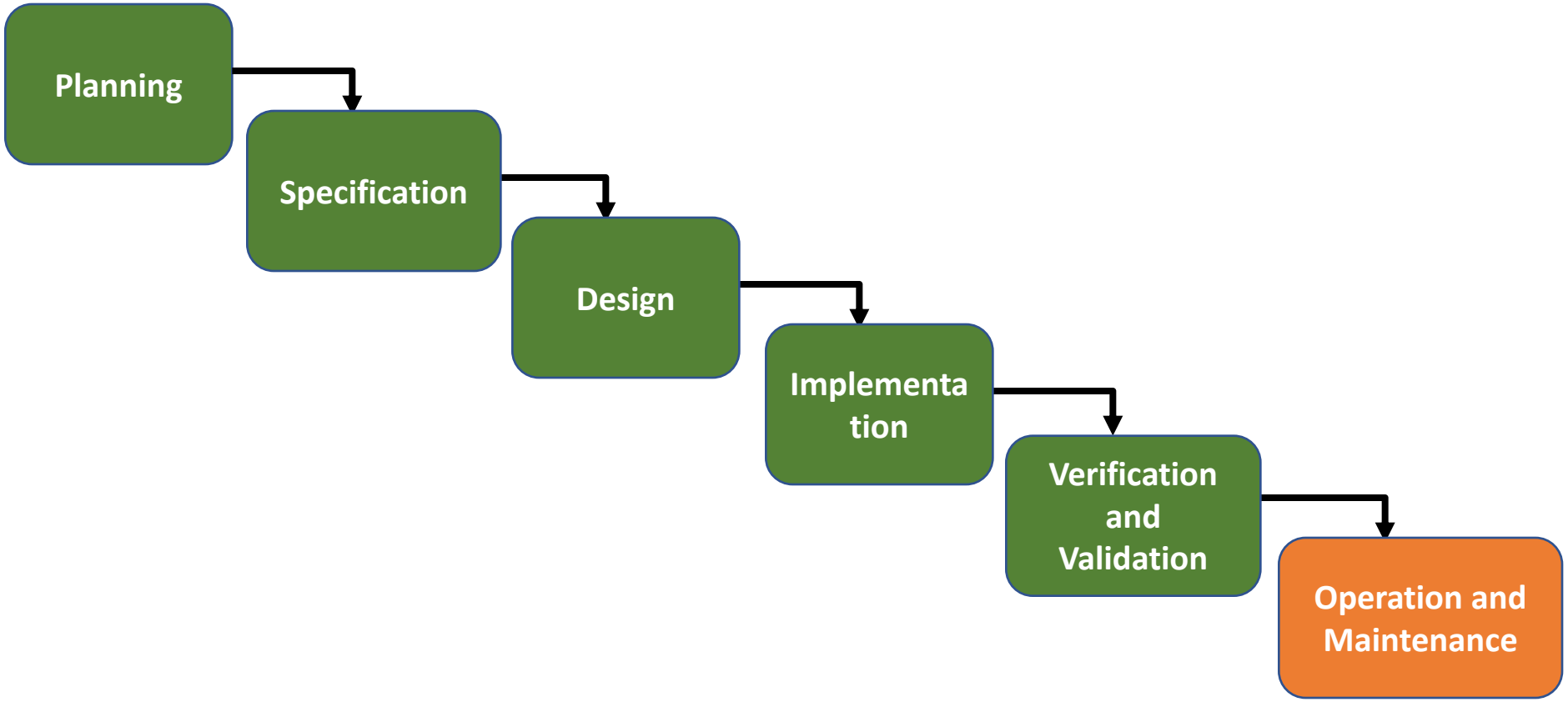
Chapter 6 - Acceptance, Introduction, and Maintenance

Walter F. Tichy

Where ae we right now?



Content

- The acceptance phase
- The introduction phase
- The maintenance phase
- Maintenance improvement

The acceptance phase

- The completed overall product is accepted and introduced to the user, i.e., put into operation.
- From this point on, the product is subject to maintenance

The acceptance phase

- Activities
  - Delivery of the complete product including all documentation to the client
  - An acceptance test is generally associated with the delivery.
  - In an acceptance test it is also useful to perform load or stress tests
  - The result of the acceptance test is an acceptance protocol.

The acceptance phase

- Acceptance
  - After successful testing of the product,
  - the client signs a formal acceptance declaration (in the legal sense)
  - this may include lists possible alterations that are still necessary.

The acceptance phase

- Acceptance test also depends on whether the customer ...
  - only uses, but does not maintain the product
  - uses and maintains it
  - Which alternative the client chooses should already be known when the contract is awarded
    - The quality attributes relevant to the client depend on the alternative chosen.

The acceptance phase

- Product usage
  - Key quality attributes:
    - Usability
    - Integrity
    - Efficiency
    - Correctness and
    - Reliability

- Maintenance
  - Additional quality attributes:
    - Maintainability
    - Testability
    - Flexibility

The introduction phase

- Activities
  - Product installation
    - Setting up the product in its target environment for the purpose of operation
  - Training of users and operating personnel
    - After installation, users must be trained in operating the product
  - Commissioning of the product
    - Transition between installation and operation

The introduction phase

- Introduction protocol
  - All incidents that occur during the introduction phase are recorded
- Introduction must be carefully planned
- Treat large-scale product launches like innovation launches
  - Help users overcome fear/rejection of new things
  - Identify champions/enthusiasts and let them help to convince others.

The introduction phase

- When replacing an existing system: Changeover
  - Plan the necessary steps in a network plan
  - Important task:
    - Converting existing databases to new formats
    - Manual card indexes and paper files must be compiled and prepared before they can be captured for the new data management system
    - For extensive inventories
      - Time must be scheduled for scanning documents and manual data entry

The introduction phase

- Changeover
  - The biggest problem arises when transferring "live" data sets, e.g., warehouse files, customer data in a running system
    - changeover must happen at a certain time or at several times
  - Legacy data requires conversion programs, which must be developed
  - Consideration should also be given to how to verify the accuracy of the data sets created.

The introduction phase

- Commissioning possible in 3 ways:
  - Direct transition
  - Parallel run
  - Test run

The introduction phase

- Direct transition
  - There is an immediate transition from the old to the new system
  - The use of the old system is stopped in order to put the new system into operation immediately
  - A weekend or a holiday period is chosen for the changeover work
  - Direct transition without further precautions is risky
- If problems occur, be prepared to go back to the old system until the problems are solved.

The introduction phase

- Parallel run
  - The transaction data is processed in both the old and the new system so that the results can be compared with each other
  - Advantage:
    - One has safety in case the new system does not work
  - Disadvantages:
    - High cost: data must be entered twice
    - Difficulties caused by running two systems in parallel.

The introduction phase

- Test run
  - First possibility
    - New system works with data from past periods so results are known and can be verified
    - During the test runs, users report problems and failures
    - Current processing takes place in the old system
    - When new system is running smoothly and is up-to-date, change over directly.
  - Second possibility
    - Introduction of the new system in individual stages by successively taking over different functional areas
    - This also facilitates the transfer and training of personnel.

The introduction phase

- Pilot installation or beta test
  - If a software product is produced for the anonymous market, a series of pilot installations at pilot customers (beta test) are carried out before a general sales release.
  - Problems are fixed during this period
  - Training videos or tutorials should be prepared for the official role-out.

The introduction phase

- End of product development
  - After successful introduction of the product, the official release of the product takes place
  - This concludes the product development
- Acceptance & Introduction Phases produce the following results:
  - Total product incl. total documentation
  - Acceptance protocol
  - Introduction protocol.

The introduction phase

- As a basis for subsequent maintenance, it is necessary to archive all products
  - The maintenance archive must keep different versions of each product.
  - The maintenance archive must include information about the installed versions at the individual customers
  - Or the software identifies itself.
  - Should automatic updating of the software take place?

The maintenance phase

- Maintenance
  - Starts with the successful acceptance and introduction of a software product
  - This is a somewhat arbitrary point in time, as the software may have to be extended or adapted after the initial delivery.

The maintenance phase

- After commissioning a product...
  - failures occur in daily operation
  - environmental conditions change
    - new system software (OS; DB)
    - new hardware
    - new organizational embedding
  - new wishes and requirements arise
    - new functions
    - changed user interface
    - increased speed.

The maintenance phase

- Software aging
  - Software that is not constantly adjusted to both the changing environment and new requirements will eventually age and become obsolete
  - It can then no longer be used for the purpose originally intended
  - "Software becomes obsolete to the extent that it fails to keep pace with reality." [Sneed 83]

Maintenance: tasks and their effort

- 4 categories of maintenance & care
  - Corrective activities
    - Stabilization / correction
    - Optimization / performance improvement
  - Progressive activities (care)
    - Adaptation to changed environment or porting to other platforms
    - Extension / new or improved functionality

Maintenance: tasks and their effort

- Stabilization / correction
  - All activities that repair defects
  - These may be defects that have already entered the product during development or new defects that arise during maintenance.

Maintenance: tasks and their effort

- Stabilization / correction
  - Maintenance defects, the so-called *second level defects,* multiply particularly quickly.
    - Defects introduced during defect repair
    - They can soon make up the majority of defects
    - Cause:
      - Poor design and susceptibility to failure of the original product
      - Poor documentation
      - Lack of product understanding among maintenance personnel

Maintenance: tasks and their effort

- Optimization/performance improvement
  - Freshly deployed software often consumes more time and memory than is available to perform its tasks.
  - Optimization rarely occurs before the first release
    - Once a product is functional, it is released
  - Optimization is postponed to maintenance:
    - All activities to improve performance
      - Make a runtime profile of all methods to find those that consume most time or are called most often. These are the ones to optimize.
      - Fine optimization and reduction of storage requirements
      - introduction or optimization of parallel threads
      - In some cases, restructuring is also necessary to achieve performance improvements.

Maintenance: tasks and their effort

- Adpatation
  - Adaptations are forced by changes in the environment:
    - Changes in the technical environment
      - e.g., new system software, new devices
    - Changes in the user interfaces
      - e.g., modified windows or forms, Internet-enabling, voice commands
    - Changes in business rules
      - e.g., changes in legislation, new company regulations.
    - Porting to additional platforms

Maintenance: tasks and their effort

- Extensions
  - Lead to a functional addition to the product
  - Functions that were foreseen or planned during initial development but not implemented are now developed
  - Or new functions arise from the requirements of the operation of the software.

Maintenance: tasks and their effort

- Other classification:
- Corrective activities
  - Include the identification and correction of
    - Software defects
    - Performance deficiencies
  - This includes "emergency repairs" that must be performed immediately to maintain ongoing operations
  - Corrective implementation is also part of these activities to bring it in line with specified product requirements and deliverables.

Maintenance: tasks and their effort

- Customizing activities
  - Adapt the software to changing product environments
- Perfecting activities
  - Increase performance, improve cost-effectiveness, processing effectiveness and maintainability
  - This also includes enhancements due to new user requirements
- Most of the effort is spent on customizations and extensions
  - 60% to 80%.

Plannability of maintenance and care

Corrective Activities

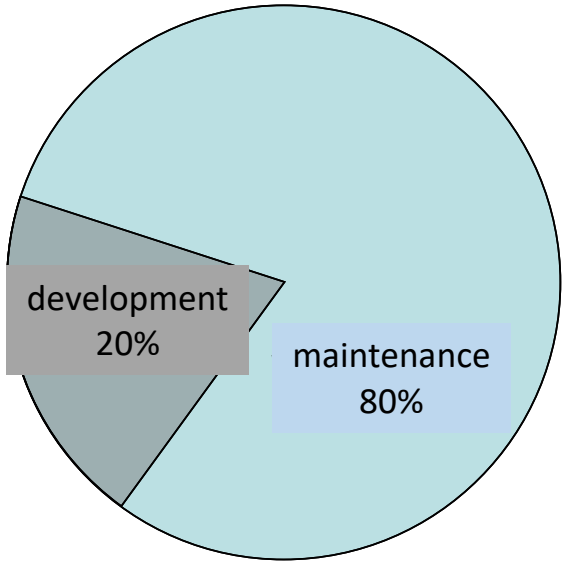
- Locate and fix defects in in-service software when failures occur.
- Is event-driven, therefore difficult to plan.

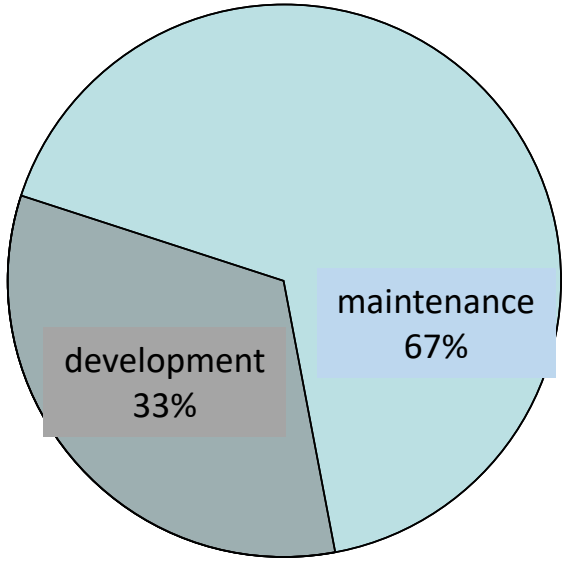
Customizing and Perfective Activities

- Localize and perform customizations, modifications, and enhancements to in-service software when the nature of the desired changes/enhancements is determined.
- Can be planned.

Maintenance vs. development

- *Life cycle of* a product
  - Effort for a product can be divided into
    - the development effort
    - the maintenance effort





Maintenance vs. Development

- Rules of thumb
  - The effort for maintenance is usually greater than the development effort
  - The maintenance & support effort is typically a factor of 2 to 4 greater than the development effort for a comprehensive product.
  - Such an effort distribution means
    - In extreme cases, out of 100 employees in a software department, 80 employees are involved in the maintenance of "old" software
    - Only 20 employees develop new software.

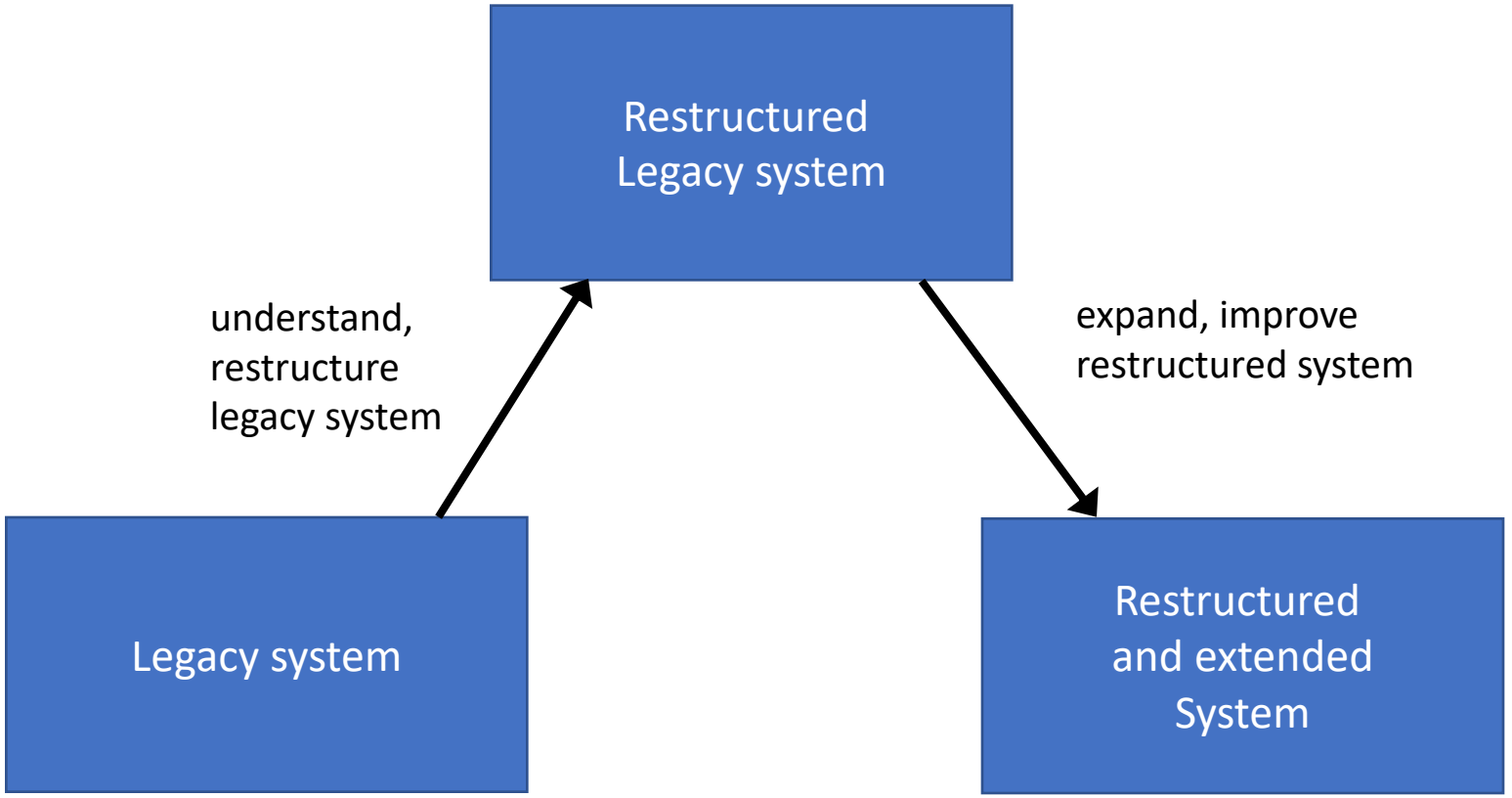
Viewing Maintenance as further Development

- Adaptations, Enhancements
  - Adaptations and enhancements of a product are characteristic for further developments or for new versions of products
  - It therefore makes sense - apart from minimal changes - to let all maintenance activities run through the normal software development process
  - In the evolutionary and incremental process model
    - No longer a maintenance phase, maintenance activities are seen as creating a new product version.

Software Re-Engineering

- Refurbishment or restoration (*re-engineering*) of legacy systems
  - There will always be old software
  - Old software is the predominant part of the installed software
  - During re-engineering, an understanding of the old software and its transformation into a more maintainable form (newer language, clearer design, modularized structure) takes place before the actual change is made
  - The development of tools that facilitate software restoration is an important area of research

The Process of Software Re-Engineering



Maintenance organization

- Should maintenance be independent?
- Or should developers also maintain?

Advantages of independent maintenance organization

- Clear allocation of maintenance and development costs
- Relieve developers from maintenance tasks and especially from parallel execution of different activities
- Qualitatively better acceptance test by the maintenance team
- Better customer service through focus on maintenance
- Hiring of specialized employees or targeted training of employees
- More efficient communication between maintenance employees
- Higher productivity through specialization and related product knowledge.

Disadvantages of independent maintenance organization

- Maintenance work can get a "bad image", which decreases the motivation of employees
- Valuable knowledge about the product is lost during the transition from development to maintenance
- Coordination problems between development and maintenance, especially when new products replace old ones.
- The developers do not have to bear the consequences of their development
- The maintenance employees have to familiarize themselves with the systems in a time-consuming manner
- It is difficult to achieve an even workload for employees.

Maintenance organization

- There is no perfect organization
- Compromise:
  - Separate organizations, but
  - employees "rotate" between the two organizational units
    - Then developers see what is important in maintenance and vice versa.
    - You don't get "stuck" in maintenance forever.
  - The success of maintenance depends less on software technology and more on organization and management.