Cornell University
Computing and Information Science

CS 5150 Software Engineering

Acceptance Testing and Delivery

William Y. Arms

Acceptance Testing

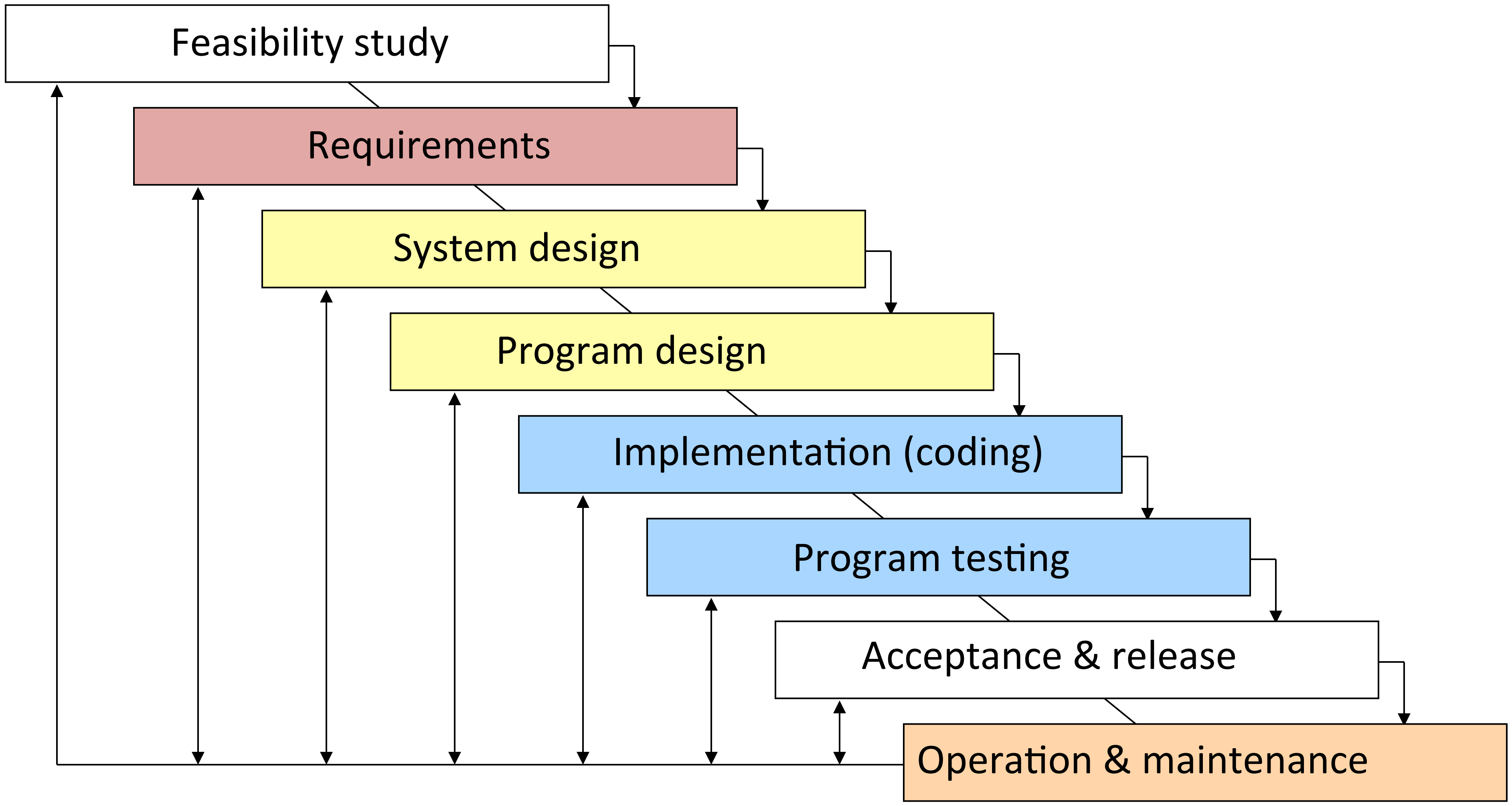
Acceptance testing

The **complete system**, including documentation, training materials, installation scripts, etc. is tested against the **requirements** by the **client**, assisted by the developers.

- Each requirement is tested separately.
- Scenarios are used to compare the expected outcomes with what the system does.
- Emphasis is placed on how the system handles problems, errors, restarts, and other difficulties.

Is the system we have built, the system that you wanted? Does it meet your requirements?

Acceptance Testing in the Modified Waterfall Model



Acceptance Testing with Iterative Development

Iterative development

If the client is properly involved in the development cycle:

- The client will have tested many parts of the system, e.g.,
    - -> At the end of each iteration
    - -> During user testing
- Problems and suggestions for improvement will have been incorporated into the system

BUT: There must still be an acceptance test of the final system before it is released.

Acceptance Testing with Incremental Development (Sprints)

Incremental development

Acceptance testing is particularly important with incremental development, since each sprint should end with a release.

- Each sprint should be a complete development process, ending with **acceptance testing** by the client and release.
- If several sprints build on each other, each sprint may need to repeat the acceptance tests for **earlier sprints** to check that they are still met.

Resources for Acceptance Testing

Acceptance Testing is a important part of a software project

- It requires time on the schedule
- It may require substantial investment in test data, equipment, and test software.
- Good testing requires good people.
- Help systems and training materials are important parts of acceptance testing.

Acceptance Tests

- **Closed box by the client** without knowledge of the internals
- The entire system is tested as a whole
- The emphasis is on whether the system meets the requirements
- The tests should use real data in realistic situations, with actual users, administrators, and operators

The acceptance tests must be successfully completed before the new system can go live or replace a legacy system.

Completion of the acceptance tests may be a contractual requirement before the system is paid for.

Techniques for Release

The transition from the previous version of a production system to a new release is challenging.

Parallel Testing:

Clients operate the new system alongside the old production system with same data and compare results

Alpha Testing:

Clients operate the system in a realistic but non-production environment

Beta Testing:

Clients operate the system in a carefully monitored production environment

Release: Parallel Testing

Parallel Testing

For data processing systems, such as financial systems, payroll, etc., the **old** and the **new** systems are run together for several productions cycles to check that the new system replicates the functionality of the old.

- Requires two sets of everything (staff, equipment, etc.).
- Requires software to control changeover (e.g., do not mail two sets of payments).
- Requires automated scripts to compare results.

Parallel testing may take several months. Often, the new system will be brought into production in phases.

Release: Alpha and Beta Testing

Alpha testing can be done with software that lacks some functionality. Beta testing requires fully functional system.

Alpha and Beta testing must be managed

If you simply make versions of the software available to clients:

- Many clients will never use it.
- Other clients will use a few features.
- Only a few clients will test it systematically.
- Only a few clients will report problems systematically.

What **incentives** can you give clients to test your software for you?

- Financial (e.g., discounts on products)
- Prestige (e.g., recognition, publicity)

Delivery: Summary

A good delivery package results in:

- happy client
- happy users
- less expense in support and maintenance

But many projects rush the packaging, help systems, and training materials, give them to the least experienced members of the team, do not test them properly, and generally neglect this part of the software process.

Delivering the System

The best is the enemy of the good

- No system is ever perfect.
- If you insist on perfection:

  -> you will never release anything

  -> you will still never reach perfection (because the ultimate test comes from the users)

Learn what is essential to satisfy the client and focus on it.

Delivering the System

- Pressures to get products to market and in operation often lead to bad decisions.
- Trade-offs must be made between the cost of packaging, future support and maintenance, and the risk of later problems.

  All are expensive, but, in my experience, the pain of being late is usually less than the pain of putting a bad system into operation.
- Different categories of software product need different packaging and delivery procedures.

Delivery of Software

Shrink-wrapped package (may be downloaded)

- Installation scripts

  -- automatic
  -- varieties of hardware and operating systems
  -- uninstall, reinstall, etc.

- Support (very expensive when it requires staff)

  -- training
  -- documentation (user, system administrator, expert user)

- Maintenance

  -- client does not have source code
  -- no bug fixing except with new release

Delivery of Software

Data processing system

Acceptance

- acceptance period may cover several months
- client should be comfortable with complete system

Support

- client should be self-sufficient
- documentation and training for system administrators and operators
- well organized source code for maintenance
- maintenance and support contracts

Delivery of Software

Embedded system

Acceptance

- hardware and software developed together
- acceptance tests are a combination

Maintenance

- bug fixes may require servicing the hardware
- errors may be expensive or dangerous

Support

- training for support personnel
- documentation and training for users

Training

Time and money spent on training is usually well spent:

- one-on-one
- in-house training
- training courses
- distance education
- online tutorials

Development team provides information for training materials:

- users (perhaps several categories)
- system administrators
- system maintainers
- trainers

Training and Usability

A well-designed system needs less training

- good conceptual model
- intuitive interfaces

Different skill levels need different types of training

- skilled users work from the conceptual model
- less-skilled users prefer cookbook sets of instructions
- occasional users will forget complex details, but remember general structure

Help Systems

Resources

- A good help system is a major sub-project (time-consuming, expensive)
- A good help system saves user time and support staff (time-saving, cost-saving)

Help system design

- Users need many routes to find information (index by many terms, examples, mini-tutorials, etc.)
- Help systems need to be tested with real users

Categories of Documentation

Software development

- Requirements, design
- Source code, test plans and results

User

- Introductory (various audiences)
- User manual
- Web site of known problems, FAQ, etc.

System administrator and operator

- System manuals

Business

- License, contract, etc.

Documentation

Online documentation

- Much cheaper than print
- Fewer restrictions on numbers of pages, colors, etc.
- Easy to update (e.g., over the Internet)

but... Cannot be used if the user's system is down

Installation Tools

Creating installation scripts may be a major sub-project

- Different scripts, tools and procedures for different categories of software
- Testing must be extensive with real users in their own environment

CS 5150: What Materials Should You Deliver?

When you leave, all that the client has is your software and your documentation. Imagine that you work for the client and are asked to take over this system. What would you want?

Different projects will have different deliverables

Discuss delivery with your client

- Place your materials on a project web site.
- Materials can be in any format, need not be huge, but must be current.

Delivery: Check List for CS 5150 Projects

Documentation

- Requirements, updated to reflect delivered system
- System & program design, updated to reflect delivered system
- Instructions for: users, administrators, operators
- Presentation slides, updated to reflect delivered system
- Business documentation, e.g., copyright license

System

- Source code and matching binary for all programs
- Installation scripts, etc.
- Test scripts, test data, and test reports

Delivery: Maintenance

Most production programs are maintained by people other than the programmers who originally wrote them.

(a) What factors make a program easy for somebody else to maintain?

(b) What factors make a program hard for somebody else to maintain?