Loose ends

Integration and System Tests

Ragnhild Van Der Straeten - ULB - Software Engineering and Project Management - 2012/2013

Integration and System Testing

- **Integration:** Building a (partial) system out of the different modules. Integration proceeds by iterations.
- **Builds:** A build is a partial system made during integration. An iteration may involve several builds.
- **Associated tests:** interface tests, regression tests, integration tests, system tests, usability tests, acceptance test.

Planning Integration

- Identify parts of architecture that will be integrated in each iteration:
  - Try to build bottom-up (no stubs for lower levels).
    - Document requirements and use cases supported by iteration.
    - Retire risks as early as possible.
- Plan inspection, testing and review process.
- Make schedule.

Testing during Integration

- **Retest** functions, modules in the context of the system (e.g. using no or higher level stubs).
- **Interface testing** of integration.
- **Regression tests** ensures that we did not break anything that worked in the previous build.
- **Integration tests** exercise the combination of modules, verifying the architecture (and the requirements).
- **System tests** test the whole system against the architecture and the requirements.
- **Usability testing** validates the acceptability for the end user.
- **Acceptance testing** is done by the customer to validate the acceptability of the product.

Integration Test Road Map

- Plan integration.
- For each iteration:
  - For each build:
    - Perform regression tests from previous build.
    - Retest functions, classes, modules.
    - Test interfaces.
    - Perform integration tests.
  - Perform iteration system and usability tests.
- Perform installation test.
- Perform acceptance test.

Integration Testing

- Decide how and where to store, reuse, code the integration tests (show in project schedule).
- Execute unit tests in context of the build.
- Execute regression tests.
- Ensure build requirements and (partial) use cases are known.
- Test against these requirements and use cases.
- Execute system tests supported by this build.

Interface Testing

- When testing integrated components or modules: look for errors that misuse, or misunderstand the interface of a component:
  - Passing parameters that do not conform to the interface specification, e.g. unsorted array where sorted array expected.
  - Misunderstanding of error behaviour, e.g. no check on overflow or misinterpretation of return value.

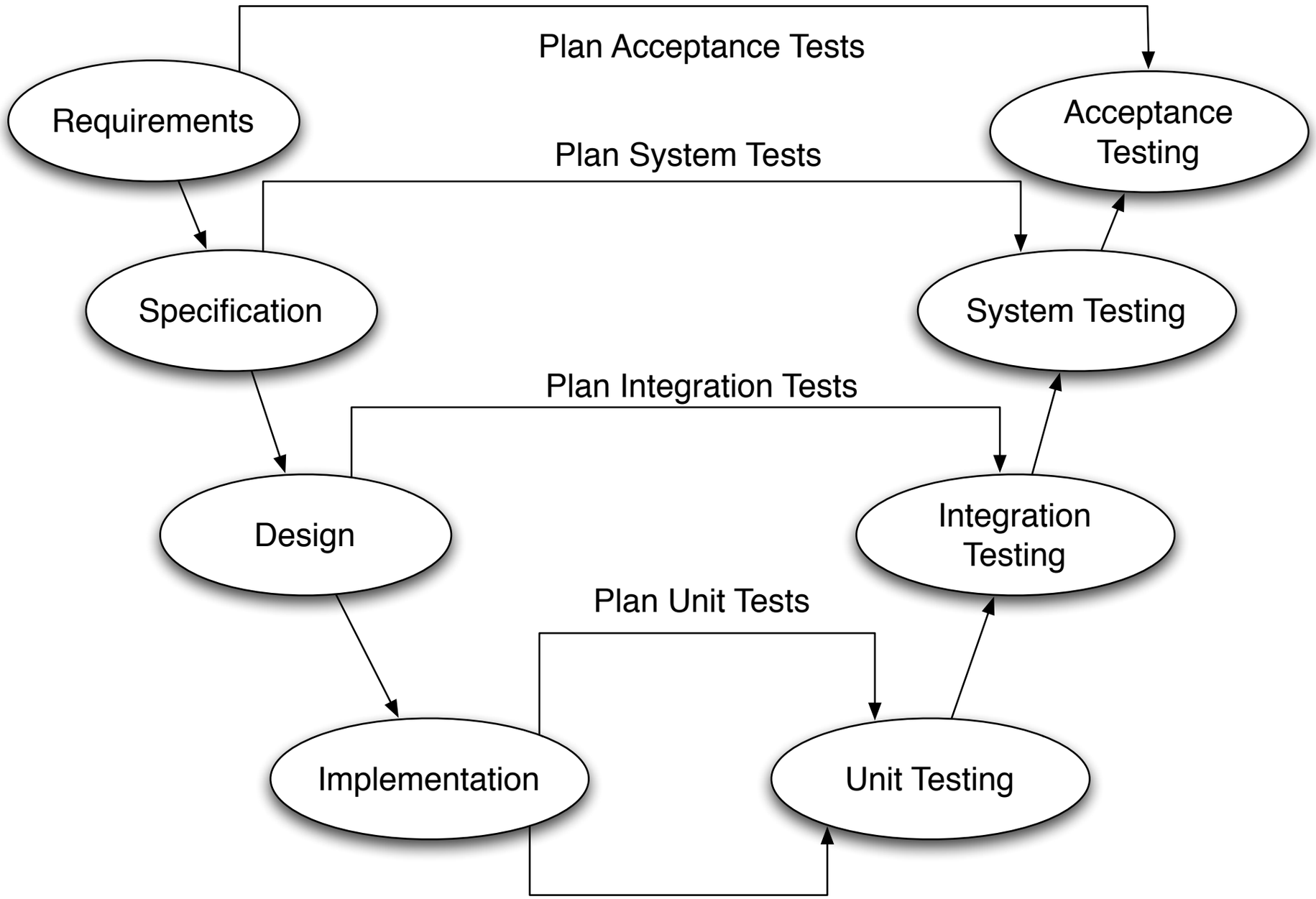
System Testing

- A test (script) for each requirement/use case.
  In addition, do tests for:
  - High volume of data.
  - Performance.
  - Compatibility.
  - Reliability and availability (uptime).
  - Security.
  - Resource usage.
  - Installability.
  - Recoverability.
  - Load/Stress resistance.

Usability Testing

- Against requirements.
- Typically measured by having a sample of users giving a score to various usability criteria.
- Usability criteria should have been specified in advance in the SRS.

Traditional Elements of a Test Process



Inspections, process metrics and process improvement

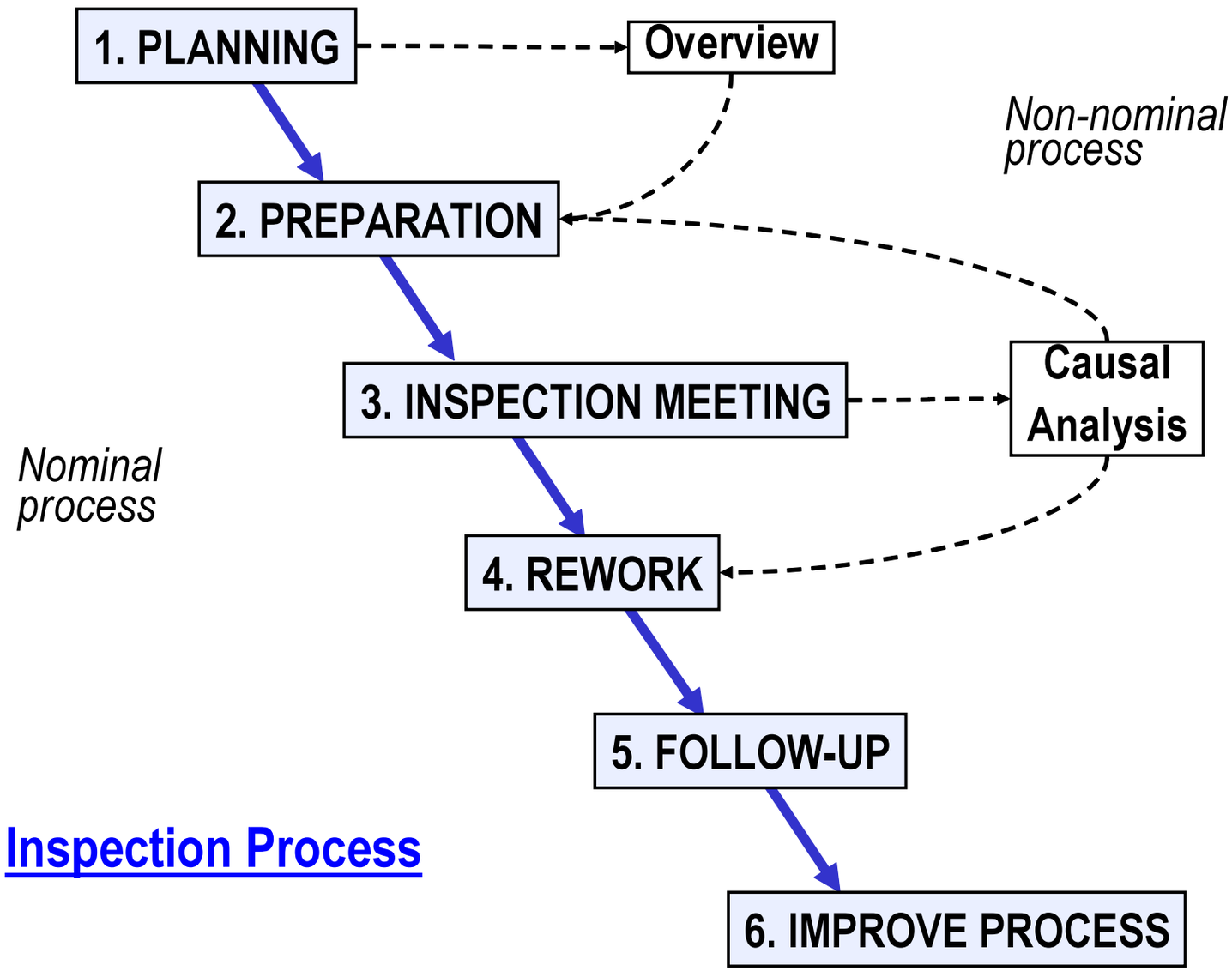
Inspections

- inspection is
  - a quality technique that focuses on reviewing the details of a project artifact in an organised and thorough manner.
    - performed periodically during all software engineering phases.
  - **purpose**: to assure the artifact's correctness by seeking defects.
  - meeting of inspectors is held at which defects are identified.

Inspection Principles

- Defect **detection** only.
- Peer (not supervisor-subordinate) process.
- Only **best effort** of author should be examined.
- Specified roles:
  - Moderator (is also inspector).
  - Author (is also inspector, answers questions)
  - Reader (is also inspector): leads team through the work.
  - Recorder (is also inspector).
- Inspectors should **prepare** the inspection.

Inspection Process



Inspection Process

Example

Inspecting requirements.

faulty:
If the temperature is within 5.02% of the maximum allowable limit, as defined by standard 67892, then the motor is to be shut down.

Correct:
If the temperature is within 5.02% of the maximum allowable limit, as defined by standard 67892, then the motor is to be powered down.

! "shut down" /= "power down"
! Very expensive to find and fix after implementation.

One way to prepare and conduct inspections

- Build inspections into schedule (time for preparation, meeting).
- Prepare for collection of inspection data.
  - Number of defects/KLOC, time spent.
  - Form, e.g. with description, severity.
  - Who keeps inspection data, usage of . . .
- Assign roles. E.g. author, moderator/recorder, reader or, minimally, author/inspector.
- Ensure that each participant prepares: bring filled defect forms to meeting.

Defect Tracking

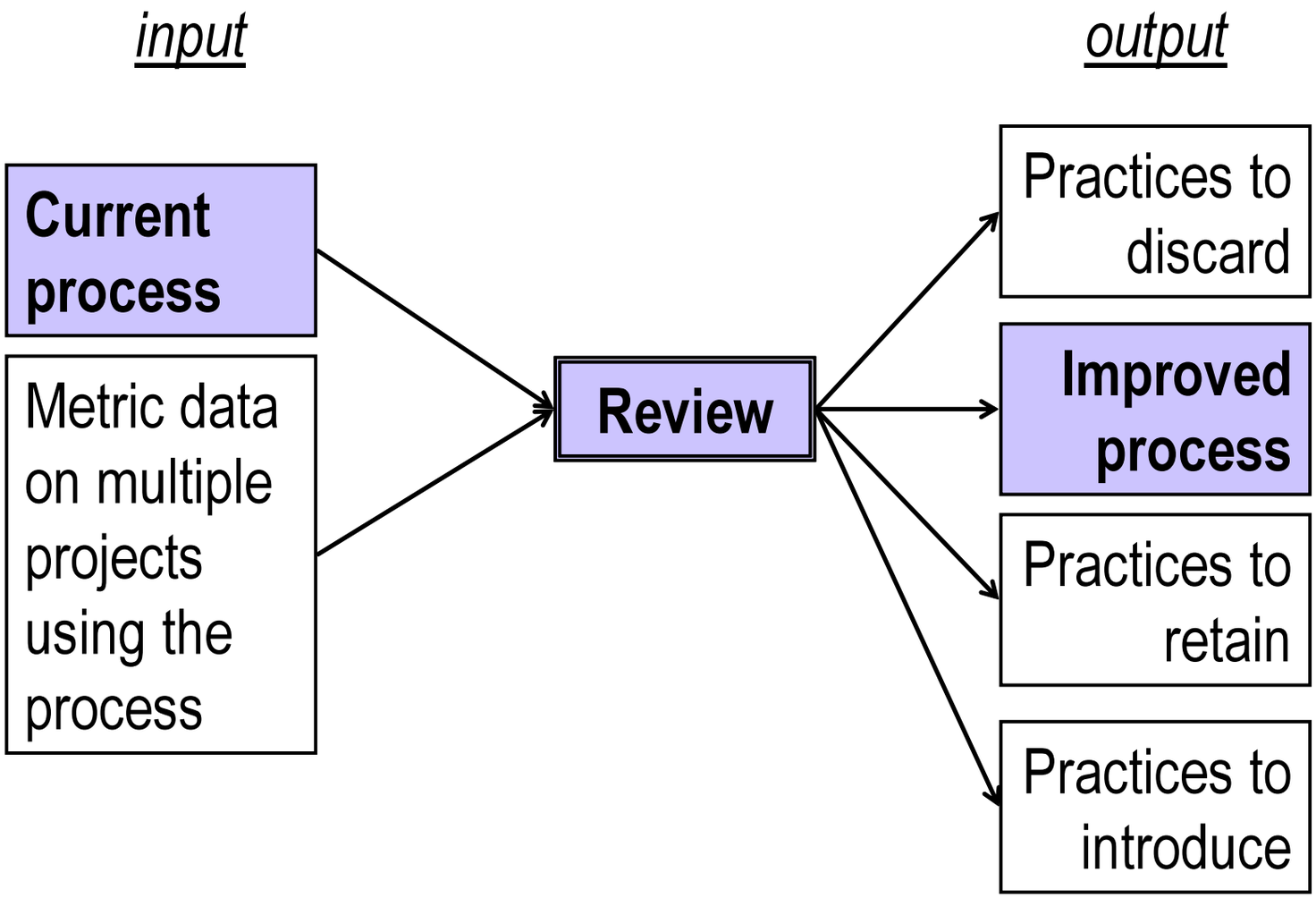
Defect Tracking								
Name	**Description**	**Discov-ering enginr.**	**Respo n-sible enginr.**	**Date opene d**	**Source**	**Seve -rity**	**Type**	**Status**
Check-out flicker	Checkout screen 4 flickers when old DVDs are checked out by hitting the Checkout button.	Kent Bain	Fannie Croft	1/4/04	Inte-gration	Med	GUI	Being worked begun 2/10/04
Bad fine	Fine not correct for first-run DVD's checked out for 2 weeks, as displayed on screen 7.	Fannie Croft	April Breen	1/4/06	Re-quire-ments	High	Math	Not worked yet
…	…	…	…	…	…	…	…	Tested with
…	…	…	…	…	…	…	…	Resolved
…	…	…	…	…	…	…	…	…

Quality in Process Management

- Establish process metrics and targets.
- Collect data.
- Improve process, based on data.

The Meta-Process

The *Process Improvement* Meta-Process



Example Process Metri

- Number of defects per KLOC detected within 12 months of delivery.
- Variance in schedule on each phase: $(duration_{actual} - duration_{projected})/duration_{projected}$
- Variance in cost : $(cost_{actual} - cost_{projected})/cost_{projected}$
- Total design time as % of total programming time.
- Defect injection and detection rates per phase. E.g. "one defect in requirements detected during implementation".

Example Process Metri

- Time spent
  - detailed design (extra members..)
  - coding
  - self-inspection
  - unit testing
  - review
  - repair
- Defects
  - Severity: major (requirements unsatisfied), trivial, other.
  - Type (see quality).
  - Source: requirements, design, implementation.

Measure Process Effectiveness

Process →	Waterfall	Waterfall + Incremental	Pro-cess U	Pro-cess V
Average over 10 projects:				
Major defects identified within first 3 months per 1000SLOC in delivered product	1.3	0.9	0.7	2.1
Development cost per detailed requirement	$120	$100	$85	$135
Developer satisfaction index (1 to 10=best)	4	3	4	3
Customer satisfaction index (1 to 10=best)	4	6	6	2
Cost per maintenance request	$130	$140	$95	$165
Variance in schedule on each phase: $100 \times \frac{\text{actual duration - projected duration}}{\text{projected duration}}$	+20%	+70%	-10%	+80%
Variance in cost:: $100 \times \frac{\text{actual cost - projected cost}}{\text{projected cost}}$	+20%	+65%	-5%	+66%
Design fraction: $\frac{\text{total design time}}{\text{total programming time}}$ Humphrey: Should be at least 50%.	23%	51%	66%	20%

Requirements Document: 200 detailed requirements	Meeting	Research	Execution	Personal Review	Inspection
Hours spent	0.5 x 4	4	5	3	6
% of total time	10%	20%	25%	15%	30%
% of total time: norm for the organization	*15%*	*15%*	*30%*	*15%*	*25%*
Self-assessed quality 1-10	2	8	5	4	6
Defects per 100	N/A	N/A	N/A	5	6
Defects per 100: organization norm	*N/A*	*N/A*	*N/A*	*3*	*4*
Hours spent per detailed requirement	0.01	0.02	0.025	0.015	0.03
Hours spent per detailed requirement: organization norm	0.02	0.02	0.04	0.01	0.03
Process improvement	**Improve strawman brought to meeting**		**Spend 10% more time executing**	**Project Metric Collection for Phases**	
Summary	**Productivity: 200/22 = 9.9 detailed requirements per hour**				

Capability Assessment

- **CMM1 Initial**: undefined ad-hoc process, outcome depends on individuals (heroes).
- **CMM2 Repeatable**: track documents (CM), schedule, functionality. Can predict performance of same team on similar project.
- **CMM3 Defined**: CMM2 + documented standard process that can be tailored.
- **CMM4 Managed**: CMM3 + ability to predict quality & cost of new project, depending on the attributes of its parts, based on historical data.
- **CMM5 Optimized**: CMM4 + continuous process improvement, introduction of innovative ideas and technologies.