Ch17. Validating the Requirements (P329 – P349)

Validation and verification: fourth component of requirements development. RV is designed to ensure that:

1. Software requirement could describe the intended system capabilities and properties that will satisfy the various stakeholder’s needs.
2. The software requirements are accurately derived from the business requirements, system requirements, business rules, and other sources.
3. The requirements are complete, feasible, and verifiable.
4. All requirements are necessary, and the entire set is sufficient to meet the business objectives.
5. All requirements representations are consistent with each other.
6. The requirements provide an adequate basis to proceed with design and construction.

Reviewing requirements:

1. Informal review: useful for getting glaring errors, inconsistencies, and gaps, making your project meet the required characteristics. However, they are not systematic, thorough, or performed in a consistent way. That is, for everyone who participated in the feedback may has his/her own interpretation on some ambiguous statements. Methods of informal review could include:
2. peer desk check: ask one colleague to look over your work
3. pass around: invite serval colleagues to examine a deliverable concurrently
4. walkthrough: ask author to describes a deliverable and solicits comments
5. Formal peer review: The member of a formal review team share responsibility for the quality of the review,. Methods include:
6. Inspection: inspection of requirements documents, useful for maximizing the quality of software. However, it could be time-consuming and tedious.
7. Risk analysis: alternative when time is limited.

The inspection process:

The inspection involves a small team of participants to carefully examine a work product for defects and improvement opportunities. It sets a quality gate that project deliverables must pass before they are baselined.

1. Participants: you should have all necessary people in an inspection meeting. The participants should represent four perspectives:
2. The author of the work product and perhaps peers of the author: finding who could find requirements-writing errors to look for
3. People who are the sources of information that fed into item being inspected: they could be actual user representatives to the author of a predecessor specification, or product champions, to ensure that the requirements describe their needs correctly and completely.
4. People who do work based on the item being inspected: A tester is most likely to catch an unverifiable requirement; a developer can spot requirements that are technically infeasible.
5. People who are responsible for interfacing systems that will be affected by the item being inspected.

Try to limit the team to seven or fewer inspectors, and the author’s manager normally should not attend an inspection meeting, unless the manager is actively contributing to the project and his presence is acceptable to the author since the inspection would reveals defect of the project and which would lead a bad impression of the author, also, the presence of the manager may stifle discussion from other participants.

1. Inspect roles: all participants in an inspection, including the author, look for defects and improvement opportunities.
2. Author: created maintains the work product being inspected. The author usually leads the discussion, if not, the author should just listen to the comments.
3. Moderator: plans the inspection with the author, coordinates the activates, and facilities the inspection meeting.
4. Reader: paraphrases the requirements and model elements being examined one at a time, the other participant then point out potential defects and issues that they see.
5. Recorder: document the issues raised and the defects found during the meeting. The recorder should review aloud or visually share what he wrote to confirm its accuracy.
6. Entry criteria: it set some clear expectations for authors to follow while preparing for an inspection.
7. Standard template and avoid any spelling, grammatical, or formatting issues.
8. Line numbers or other unique identifiers are printed on the documents to facilitate referring to specific locations.
9. All open issues are marked as TBD or accessible in an issue-tracking tool.
10. The moderator did not find more than three major defects in a ten-minute examination of a representative sample of the document
11. Inspection stage:

图示

描述已自动生成

1. Planning: the author and moderator plan the inspection together. They determine who should participate, what materials the inspectors would receive prior to the inspection meeting, and total meeting time needed to cover the material, and when the inspection should be scheduled. The number of pages reviewed per hour has a large impact on how many defects are found. Adjust the rate based on:
2. The team’s previous inspection data, showing inspection effectiveness as a function.
3. The amount of text on each page
4. The complexity of the requirement
5. The likelihood and impact of having errors remain undetected
6. How critical the material being inspected in to project success..
7. The experience level of the person who wrote the requirement
8. Preparation: the author should share background information with inspectors so they understand the context of the items being inspected and know the author’s objectives for the inspection.

图示

描述已自动生成Up to 75% of the defects found by an inspection are discovered during preparation. Plan on spending at least half as much time on individual preparation as is scheduled for the team inspection meetings.

Do not proceed meeting unless all participants have examined the work product on their own. Ineffective meetings can lead to the erroneous conclusion.

1. Inspection meeting: during a meeting, the reader lead the other inspectors through the document, describing one requirement at a time. Other inspectors bring up possible defects and other issues, the purpose of the meeting is to identify as many major defects as possible, but the meeting should not be more than two hours.
2. Rework: the author should plan to spend some time reworking the requirements to resolve the ambiguities.
3. Follow-up: the moderator or a designated individual works with the author to ensure that all open issues were resolved and that error were corrected properly.
4. Exiting criteria: must be satisfied before the moderator declares the full inspection process complete:
5. All issues raised during the inspection have been addressed
6. Any changes made in the requirements and related work products were made correctly
7. All open issues have been resolved, or each open issue’s resolution process, target date, and owner have been documented.

Defect checklist: serves as a reminder (P339 fig 17-4)

Requirements review tips:

1. Plan the examination: focus on specific section of document
2. Start early: start reviewing sets of requirement when they are on 10% completed.
3. Allocate sufficient time: give reviewers enough time to perform the reviews
4. Provide context: give reviewers context for the document and perhaps for the project if they are not all working on the same project.
5. Set review scope: use a check list
6. Limit re-reviews: do not ask anyone to review the same material more than three times
7. Prioritize review areas: prioritize for review those portions of the requirements that are of high risk or have functionality that will be used frequently.

Requirements review challenges:

1. Large requirements: perform reviews throughout requirements development. Identify high-risk areas that need a careful look through inspection, and use informal reviews for less risky material.
2. Large inspection teams: large review teams increase the cost of the review, make it hard to schedule meetings and reach agreement.
3. Make sure every participant is there to find defects, not to be educated or to protect a position
4. Understand which perspective each inspector represents.
5. Establish serval small teams to inspect the requirements in parallel and combine their defect lists.
6. Geographically separated reviewers: provide alternative to traditional review meeting. Use word-processor feature to insert their comments into the text.
7. Unprepared reviewers:

Prototyping requirements: since it is hard to visualize how the system work from reading requirements. Prototyping ensuring stake holders have share understanding of the requirements.

1. Paper mock-up: simple way
2. Proof-of-concept: demonstrate that the requirements are feasible.

Testing the requirements: designing tests will reveal many problems with the requirements long before you execute those test on tunning software.

图示

描述已自动生成

1. Business requirements:
2. Use cases:
3. Functional requirement
4. Dialog map
5. Test:

Validating requirements with acceptance criteria: customer need to assess whether a system satisfies its predefined acceptance criteria.

1. Acceptance criteria: working with customer to develop acceptance criteria provides a way to validate both the requirements and the solution itself. Acceptance criteria define the minimum condition for an application to be considered business ready. Defining acceptance criteria is more than just saying that all the requirements are implemented, or all the tests passed, it could be:
2. Specific high-priority functionality that must be present and operating properly before the product could be accepted and used.
3. Essential non-functional criteria or quality metrics that must be satisfied.
4. Remaining open issues and defects.
5. Specific legal, regulatory, or contractual conditions
6. Supporting transition, infrastructure, or other project requirements.

These could also think about the “rejection criteria” and take care for conflicting acceptance criteria.

1. Acceptance tests: creators of acceptance tests should consider the most performed and most important usage scenarios when deciding how to evaluate the software acceptability. Automate the execution of acceptance tests whenever possible, this makes it easier to repeat the test. And some acceptance testing might be performed manually by users. The tests used in user acceptance testing (UAT) should be executed after a set of functionality is believed to be release-ready. Also, notice that writing requirements are not enough, you need to make sure the requirements are right and good enough to serve as a foundation for project.