When Standards and Best Practices are Ignored

George Jackelen
EWA, Inc.
1000 Technology Drive
Fairmont, WV 26554
(304) 367-8252
gjackele@ewa.com

Mary Jackelen
Something Buy Mary
7 Hutchinson Drive
Fairmont, WV 26554
(304) 367- 1367

Abstract

One purpose for having standards and best practices is to illustrate what experience has shown to be a preferred sequence of events. For example, experience has shown it best to perform requirements analysis prior to doing design, design should be done prior to doing software coding, and software coding should be done prior to systems testing. This does not mean each event must be completed prior to going to the next event, nor that the sequence is absolute. However, experience has shown that when the identified sequence is violated (without a justified technical or business reason), risk and cost can increase to the point where the overall effort may fail.

This paper presents an actual situation where the standard software model of events and best practices are ignored and the resulting complications. Another problem was late insertion of a procedural requirement that was misunderstood.

Keywords: best practices, capabilities, features, IV&V, requirements, test cases

Introduction

A program is currently developing a system for use throughout the world. The United States (US) is the sole developer and has established protocols to link with systems from other countries. Full development of the hardware, software and user procedures is expected to take about 10 years. Different parts of the system are scheduled to come on-line, at different times, for operational testing and use.

The system will basically:

- Collect data (e.g., tens and hundreds of gigabytes a day).
- Archive the raw and processed data.
- Make the data available to subscribers.
- Allow subscribers to develop algorithms to be run against the data for analysis and to be used as input to simulators (e.g., prediction models).

Subscribers include governments, academia, industry and individuals. Subscribers may be charged for some of the work done by the system.

The development effort was contracted out based on a competitive bidding process. The developer is responsible for requirements analysis through testing and delivery. At times, developer staffing has consisted of about 750 people. System maintenance will be handled later through a separate bidding process. Customer personnel (and maybe a contractor) are scheduled to operate the system and are currently in training. The contract has no software development standards, nor detailed deliverable descriptions.

Due to the complexity, risk and cost of the effort, a separate contract was awarded to perform Independent Verification and Validation (IV&V). Part of the IV&V team was located in the vicinity of the developer (local IV&V team) and the rest of the IV&V team was located several hundred miles away (remote IV&V team).

Before going into the main topic of this paper, a real example will illustrate problems faced by the customer, developer and IV&V.

Requirements analysis

The customer provided some high-level requirements, placed these requirements under customer control and

0-7695-0068-4/99 $10.00 © 1999 IEEE

asked the developer to create lower-level requirements. The lower-level requirements consisted of:

- A repeat of the customer requirements divided by incremental system releases. Due to the system's interaction with other systems/missions, part of the customer's system has to be on-line early to support these missions. As a result some customer requirements had to be clarified (i.e., not changed) to indicate what requirement parts would be required for what mission. In some cases the clarification included: "For mission X, this will be done manually. Full automation will not occur until mission Y". In other cases the clarification was an interpretation of the requirement to satisfy mission X criteria, and then a second interpretation to satisfy mission Y criteria.
- Developers (mainly software coders) then created code requirements. Many times these requirements were based on what was coded rather than being derived from higher-level requirements.

During requirements analysis, IV&V was tasked to verify:

- Traceability of the requirements. A tool was acquired by the developer to provide pointers for the parent and children of each requirement. IV&V then ensures pointers existed (physical traceability) and the resulting traceability is logical. Thus, physical traceability consists of ensuring a pointer exists to a parent/customer requirement and to a child requirement (as usual, this is ignored for the lowest- and highest-level requirements). Logical traceability is much harder to perform due to:

- The way the code requirements were created.
- A major breakdown in communications between the developer and IV&V (developer was highly sensitive to IV&V comments and the fear IV&V would expose problems to the customer).
- Developer did not provide answers to IV&V questions, e.g., how does a set of code requirements relate to a higher-level requirement.
- The remote IV&V team had the main responsibility to perform this analysis and they had the least knowledge of the system being developed and the day-to-day events of the development.
- Completeness of the requirements. For the same reasons as above, this was also very difficult. This task required verifying if lower-level requirements completely satisfy higher-level requirements. Lack of system knowledge by the remote IV&V team and the inability to determine what the coders were thinking when they developed the code requirements caused a wide range of IV&V comments. For example, some IV&V evaluators would assume the coders knew what they were doing, while other IV&V evaluators always wanted more details, e.g., is output a hardcopy or electronic.

Even though many of the IV&V comments were valid, the comments were ignored by the developer who continued to work according to their schedule rather than spending time correcting valid IV&V identified problems. For example, one IV&V comment was the code requirement had a maximum response time greater than what was stated in the higher-level requirement. Even though this was probably a typing error, the error is not fixed after 18 months. IV&V discovered physical traceability errors also were not fixed, even when IV&V provided the solution. The customer has refused to become involved.

IV&V papers and briefings to the customer about these problems and the consequences were largely ignored. To illustrate the problems the program faces, the program's first major release may be delayed over a year because the user (mission customer) of the release has rejected a major portion of the product (being developed by an associated developer). As a result, this is the first customer mission to be delayed due to software problems.

Testing

The program is currently in the testing phase (even though requirements analysis is not complete) for its first major release. Over two years ago (early in the so-called design activity), the developer produced a test plan which was accepted by the customer over IV&V objections. IV&V objections were based on document inconsistencies, incompleteness and not addressing best practice issues normally presented in a test plan as presented by most standards. The test case documents were also accepted by the customer, even though IV&V had the same objections it had with the test plan.

Currently IV&V is reviewing test cases to be used during the systems and acceptance tests. Documented IV&V test case comments on what was happening include:

- Test cases are incomplete and do not verify the indicated requirements.
- Some test cases only partially verify a requirement. When this happens, test cases reference other test cases that have been identified to complete the requirement's verification. Upon examination, IV&V found several of the references were wrong or there were still parts of a requirement not covered by any test case.
- Stress testing and a test to determine if operators (not developers) could do a cold start of the entire system were not addressed by any test case.
- Many test case steps were written for software people who understand Unix. Operational system operators are not intended to be software knowledgeable.
- If a tester (i.e., a person assigned to execute a test case) completed a test without an error message, the test case was usually given a grade of "pass", even if some steps were skipped (e.g., needed software tools were not available) or a step had to be re-written to match actual events.
- Testers were testing test steps, not test cases. Many testers did not have enough knowledge of the system to determine if the test case steps satisfied any of the related requirements. Most tester comments (redlines) dealt with correcting test step procedures (e.g., keyins or system responses). For example, after a test case was completed an IV&V test witness asked the tester, "What grade should this test case receive?"; the reply was "Pass since all the test steps were run and there were no errors". The IV&V test witness then asked the tester to read the requirements the test case was to verify and to identify what test case steps verified each of the three requirements. The tester was unable to do this since the test case did not verify any part of the requirements. Without IV&V intervention the developer and customer would have been notified of an erroneous test case completion/pass.
- The unwritten objective of the testers is to show a test case passed, instead of the best practice objective of testing to find and document problems.
- Testers had no standard to determine if a test case passed, failed or was a partial pass. Each tester used their-own definition. It also appears that, as the schedule became tight, more test cases were receiving a grade of "pass". Later, a definition of pass was agreed to by the developer and customer. However, during on-site testing the testers argued against the definitions because of the resulting low percentage of test cases that were passing.
- Even though a test case received a fail or partial grade, management for the developer reported a passing grade, even if a test case was not executed. Because of this the customer asked IV&V for weekly reports on developer grades for each test case.

Capabilities, features, requirements and test cases

By contract, the developer is required to map the customer's high-level requirements to test cases and thus prove the requirements are satisfied. Because of the above problems and the customer's realization the developer was falling behind schedule, the customer developed a set of requirements called capabilities to help the developer to succeed. In a typical business, a capability is an objective, not a requirement (i.e., a capability is not measurable/testable - does not have a "shall"; it is a goal), or purpose statement.

After the customer spent several staff months on this capability effort, neither the developer nor IV&V understood how the capabilities related to the requirements, nor how to map the capabilities to the test cases. When the customer pushed the developer to provide a mapping between capabilities and test cases, the developer stated this was not part of the contract, and thus was out of scope.

The customer then used another business concept call features. In business, capabilities can be subdivided into features. Features are sub-objectives, not requirements; and therefore are not always testable. However, the customer quickly began treating features as requirements and thus caused more confusion. With this thinking in mind, the customer required the features to be included in the test cases, along with the existing requirements mapped to test cases. Again, the developer fought this work as being out of scope, but soon conceded since, in the authors' opinion, progress was being slowed, there were cost overruns, and potential failures and schedule delays could now be partially blamed on the customer.

Another reason for the developer to agree with the customer occurred when the customer decided that features were a better way of focusing the test schedule to comply with the system schedule. Several of the

features were written to reflect what was needed for key product release milestones. The developer, in the authors' opinion, also realized the features were less restrictive than requirements and the determination of what was a test case passing grade was more subjective.

Besides the fact the features were objectives rather than real requirements, the following flaws were built into the customer defined features:

- Many of the features were quotes/summaries from the developer's test case objectives rather than from the capabilities.
- Wording ("show" and "demonstrate") of the customer defined features gave testers the opportunity to use the wrong verification method. For instance, for a test case on security the developer was going to use the verification method of inspection, rather than testing, to verify, for instance, that sensitive data was encrypted, versus being transmitted in clear text. The test case instructions for this inspection only required the test to examine developer created non-compliance documents to show sensitive data would not be transmitted in clear text. Most testing standards recognize four verification methods:

- Inspection is the visual, manual examination of an entity being verified and its comparison to applicable requirements or other compliance documents to determine if there are any deviations from specifications. An inspection can certify the existence of input/output devices, alarms, operator displays, etc., by physical examination. Examples include peer reviews and walkthroughs.
- Demonstration is the observation of the functional operation of an entity being verified, in a controlled environment, to yield qualitative results without the use of elaborate instrumentation, procedures or special test equipment. A demonstration can affirm operations are consistent with the concepts and approved scenarios, e.g., observing the output on a terminal screen based on an observed input.
- Testing is a procedure or action taken to determine, under predefined-real or simulated conditions, the capabilities, limitations, characteristics, effectiveness, reliability or suitability of a material, device, system or method. Test results are compared with expected outputs to determine the success of the test.
- Analysis is a technical or mathematical evaluation based on calculation, interpolation or other analytical methods. Analysis is selected when other means are not practical.

NOTE: For a customer/acquirer, testing is the preferred verification method; but cost and schedule may result in use of the other three verification methods. Inspection, demonstration and analysis may reduce cost, but they may also increase the risk the results are not conclusive enough to ensure reliability.

To resolve this conflict of interpretation of what the features meant, the customer initiated the following:

- The developer and IV&V were to create a computer system linking the capabilities, features, requirements and test cases databases. After several weeks this effort was abandoned when nobody could understand the many-to-many mappings, for instance, between capabilities and test cases.
- The customer required the developer and IV&V to independently map the features to the test cases. The customer found they could not resolve the differences, but would not initiate, nor allow, a joint effort between the customer, developer and IV&V.
- The customer asked IV&V to examine each feature and to list what was not clearly understandable about the features. IV&V's confusion about what a feature was is shown by the resulting IV&V comments ranging from "define 'operator"' to "the features are not system objectives, but test objectives".
- This raises the issue, 'Since many of the features are actually derived from the test case objectives, if the test case objectives are satisfied, the mapped features must be satisfied." One problem with this is that system-level objectives are not guaranteed to be verified by the test cases.
- Systems and acceptance test cases rely on tables/logs/databases to act as interfaces between software programs. As a result, one test case may load a table and verify the table was loaded. Another test case may read from a similar table and act upon the results. There are few test cases verifying the correctness of the interfaces, e.g., software programs use the correct format to read/write from/to tables.

Summary and conclusion

Some progress is being made due mainly to the customer providing a new (and aggressive) liaison

to IV&V. This liaison (the sixth in four years) is more outspoken than what IV&V had before. As a result, the liaison spends many hours at customer and customer/developer meetings stating what IV&V has found, the results when this IV&V advice was ignored in the past, and what IV&V predicts will happen in the future.

The developer was acquired by a company that is proud of its IS0 9001 certification and is in the process of developing and implementing procedures, corrective actions and process improvements.

In the authors' opinion, the problems will continue because:

- Customer and developer quality assurance groups have limited involvement in the program. Also, the quality assurance groups are not pushing for more involvement.
- Customer has too many people directing the activities of the developer. Many times the directions are un-coordinated, not well thought out and contradictory.
- The program is well into systems testing, even though results show the code requirements are still being developed and the customer's high-level requirements are still being clarified and interpreted by the developer.
- The developer maintaining, and changing, its own list of customer requirements while not coordinating their changes with the customer controlled list of requirements.
- Customer signs off and approves joint reviews even though the developer does not address best practice topics for review meetings. For example, during the customer approved Critical Design Review (CDR), the developer spent the "whole" time discussing its approach to use object-oriental design and how the developer would satisfy the new requirement to run on live separate (and incompatible) platforms. Normal critical design topics were minimized.
- Contractually, the program never established software standards the developer must comply with. As a result the developer prepares plans (e.g., configuration management plan) as white papers requiring no customer approval nor implementation audits.
- Customer has repeatedly shown that schedule is more important than quality or compliance with best practices. As a result, the customer accepts developer-defined criteria and results, with minimum challenge.
- The informal barriers between IV&V, customer and developer have restricted communications and knowledge, especially with the remote IV&V team. For example, IV&V was excluded by the developer and customer from attending a critical test (developer and customer called it a demonstration) that was advertised as a major "go/no go" decision point. The demonstration was approved and the developer received its first bonus award in 2 ½ years.
- Customer program managers normally hold the position for about two years. Even though the program is currently undergoing a major evaluation of it scope, the current program manager has been replaced before the two-year period.
- Test cases are announced by the developer as being corrected (i.e., agreed implementation of customer and IV&V comments) even though only the name of a test case has been changed.

The use of standards (professional, national or international) and best practices, and the enforcement thereof, would have helped this program by providing:

- A measure of product development progress.
- Criteria for interim and final product approval.
- Responsibility for various activities.
- A sequence of product deliveries so future products are built on the results of previous products.

Biography

Mr. George Jackelen has been involved with computers since 1965. He has a BS in mathematics and physics, and a MS in computer science. Upon retiring from the US Air Force (filling many diverse computer software, hardware, staff and operations roles), he has worked for several companies to support industry and US and non-US government contracts. He is currently the EWA, Inc., program manager for two IV&V contracts and has spent several years working with IS0 and IEEE standards groups. He is a Co-Project Editor for an IS0 software configuration management Technical Report (TR) and is the Project Editor for a pending IS0 software project management TR. He has been the Publications Chair for ISESS'95, '97 and '99.

Ms. Mary Jackelen's computer experience includes working for an international insurance company and two international aerospace companies. She currently has her own business (Something Buy Mary, Inc.).