**Textbook Problem 8.2**

**Question**:

What specific activities in the advanced development phase sometimes cause it to be referred to as a “risk reduction” phase? Give an example of each activity considering a real or hypothetical system.

**Response**:

The following is a list of activities in the advanced development phase that sometimes causes it to be referred to as a “risk reduction phase”. An example of each activity with regards to a real or hypothetical system is contained below each item:

1. **Elimination of Ambiguities from Initial System Requirements**

At the beginning of the advanced development phase, the System Function Specification is key in tracing back requirements to initial Operational Requirements. Through further analysis and exploration, it is important to ensure that all derived requirements are measureable, well-understood, and unambiguous. One example of how these requirements can be further refined can be accomplished by *developing contacts with prospective system users*. BAE Systems Inc. has divisions solely dedicated to understanding Customer needs and relaying that information to Systems Engineering. This leads to a much more complete set of core requirements upfront.

1. **Assessment of Component Maturity**

A real life example of how my DoD programs at work deal with assessing component maturity is through *analysis of Technical Readiness Levels (TRLs)*. These provide a measureable and quantifiable metric for current component technology going into a system design. During competitive program captures, many of our Customer’s require for our components to meet a minimum TRL level. This gives them a sense of security that the technology is matured to a certain baseline. Some contracts may even require the technology to increase in TRL level.

1. **Prototype Development**

Further risk mitigation can be implemented when developing a complex system through the creation of a system prototype. The prototype itself can lend a huge hand in identifying which components and subsystems will require additional development or testing. A real example in which I was recently involved was in the *development of a hardware/software simulation environment for missile warning detection built to mimic a compartmentalized portion of our overall system functionality*. By building this prototype, we were able to identify requirement and testability gaps located in our overall system and in our test procedures. Through construction of this system representation, we were able to save cost and schedule with regards to program execution and overall operation.

1. **Risk Assessment**

Risk assessment is one of the most critical parts of the advanced development phase, and involves identifying sources of risk, risk likelihood, and overall criticality of all system components and interfaces. In a hypothetical system, all activities involving analysis, simulation, or implementation and testing would be great examples regarding risk assessment and mitigation. On many of my program at work, *risk registers* are developed and updated as the system design progresses, with all items being prioritized and identified in order of importance.

1. **Validation Testing**

Validation testing is an important facet that will help with risk abatement – the development and conduct of test should be performed throughout the advanced development phase, with analysis and evaluation ongoing as well. One real life example that I led last week was the *development of a regression test plan to ensure the operability of last-minute software changes*. With an upcoming Formal Qualification Test (FQT) event which will be Customer-witnessed for the next month, risk mitigation has been particularly important. This software will flight tested in a few months, so it is imperative to ensure operability prior to official release.

**Textbook Problem 9.1**

**Question**:

In spite of the effort devoted to develop critical system components during advanced development, “unknown unknowns” can be expected to appear during engineering design. Discuss what contingency actions a systems engineer should take in anticipation of these “unk-unks.” Your answer should include the consideration of the potential impact on cost, schedule, personnel assignments, and test procedures. If you have knowledge of a real-life example from your work, you may use that as a basis for your discussion.

**Response**:

I am currently the Software Test Director for my program here at work, and am a few days away from entering into a Customer-witnessed Formal Qualification Test (FQT) event. I was an integral part of the advanced development portion building up to this FQT, and performed the following actions in anticipation of “unk-unks,” which were discovered throughout all of our different testing phases (better now than out in our operational environment!):

* Regression Testing between Test Phases

Many software baseline merges were happening concurrently throughout this test event, and as expected, different parts of our algorithm were broken as a result of code collisions that were unexpected (one of the most prevalent “unk-unks” in software systems engineering). I helped mitigate this risk by implementing a rolling regression test schedule and code was updated, and ensured test procedures and requirements changes matched the required verification. This added more scope to our schedule, but in the long run, increased the integrity of our product from a Customer and end-user perspective.

* Test Procedure Reviews/Updates & Requirements Traceability (Test Case Impacts)

As engineering design is happening, chances are that legacy test procedures from the predecessor system will be updated to reflect new system requirements. With the constantly changing algorithms and numerous content updates flowing down from my Customer, it was important to analyze the scope that was affected and the changes that would be required. “Unk-unks” can appear as a result of requirement holes that have been missed due to requirement revisions and real-time code changes, so linking any changes made with test procedures updates proved to be very effective.

* Customer Inclusion and Interaction

Our Customer drove the majority of our content and requirement based changes for this FQT event. One thing that I made sure I did was to engage them early on in our discussion. An in-depth analysis was performed early on in the advanced development phase regarding our proposed test case implementations for all of our requirements scenarios, and I made sure to clarify any uncertainties with them as soon as possible. An in depth discussion was had regarding the sufficiency of our test cases and their mappings with our Operational Requirements. Many times, “unk-unks” result in requirements being missed completely in test procedures – iterative verifications with our Customer helped assure me and my team that all of our requirements were present and accounted for in our documentation and Test Plan.

**Textbook Problem 9.2**

**Question**:

External system interfaces are especially important during engineering design. Using the design of a new subway system as an example, list six types of external interfaces that will require critical attention. Explain your answer.

**Response**:

Using the design of a new subway system as an example, the following are six types of external interfaces that will require critical attention. Descriptions as to why they are important are listed below:

1. Prime Power

- This external interface is a requirement for almost all complex systems. Different types of power need to be supplied to the different components and subsystems of the subway, so defining this interface is critical to the overall system operability.

1. Communication Links

- Communication must exist between different parts of the subway. This may easily include links between the individual rail cars, identification with other subways that may be operating on the same tracks, and with other systems that handle timing and route optimization.

1. Graphical User Interface – Control Station

- In a new subway, a control system must be in place in order to provide routing information to the cars and to ensure safe transportation for all passengers. In most cases, a Graphical User Interface (GUI) is developed in order for engineers to be able to visually monitor traffic and other information from the subway itself. This is normally kept external to the system.

1. Tracking and Metrics Collection

- Information regarding traffic flow, passenger load, speed, and other metrics are important points of information to have. Tracking and metrics collection can be implemented externally via communication links to systems that assimilate this information.

1. Prognostics and Diagnostics

- Self-system repair and analysis is becoming more and more popular, and metrics like Mean Time Between Failure (MTBF) are becoming more and more prominent. External interfaces to track prognostics and diagnostics of the subway will aid in maintainability and sustainability.

1. Data Storage

- More often than not, data collected for a subway is contained on a server located external to the subway. Bi-directional links between this data server must be established in an effort to both store data and receive it if necessary.

**Textbook Problem 10.6**

**Question**:

Describe the differences in objectives and operations between developmental test and evaluation and operational test and evaluation. Illustrate your points with an example of a lawn tractor.

**Response**:

The objective and overall goal in developmental test and evaluation is to ensure the satisfaction of system specifications, with an emphasis still on ensuring operational needs of the user are being met. A realistic test environment is used, and this particular type of operation is known as a “rehearsal” for operational evaluation in the future. Given that a representation of the system environment is being used, it is important to define the following: technical parameters to be measured, summary of test events and scenarios to be validated, list of all models and simulations to be used, and a description of exactly how the environment will be represented.

On the other hand, the objective and overall goal in operational test and evaluation is validation of the system design in terms of its operational requirements, rather than specific verification that it performs according to defined specifications from previous phases of the systems engineering life cycle. The primary focus of this type of test and evaluation is on operational requirements, mission objectives and overall effectiveness, and customer/user suitability. High priority items that will most likely be addressed during this testing phase include, but are certainly not limited to: new design/engineering features, environmental limitations, system interoperability, and user interfaces.

Using a lawn tractor as an example, the following types of tests would be performed during the *developmental test and evaluation* phase of systems level testing:

* Simulated environmental effects for tractor operation
* Controlled and remote-operated “mission scenarios” for all modes of operation
* Comparison of outputs from “mission scenarios” with expected results
  + For example – if lawn tractor setting = cut lawn to 1 inch height, is this seen?

On the other hand, the following types of tests would be performed during the *operational test and evaluation* phase of systems level testing:

* Actual environmental effects for tractor operation
  + Rain / Snow / Wind / Temperature (Hot/Cold ranges)
* User-operated “mission scenarios” for all modes of operation to test all design features
  + Exploration and validation of all engineering operational requirements
  + Fulfillment of user-defined operational needs from Customer themselves
* Analysis of lawn tractor user interface and overall system interaction/operability
  + Are all user controls easy to use? Does the pilot vehicle interface (PVI) make sense?
  + Are all system operational requirements that are being tested met?
* Lawn tractor efficiency, control, responsiveness to environment, and overall comfort
  + All these measured system outputs must be quantifiable