Ponder 12

Joshua Jolley

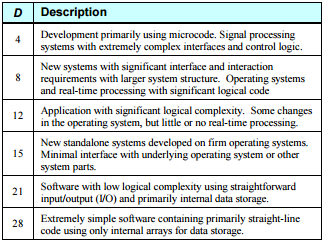
# Mobile Game

## D (Complexity)

### Define

The degree to which a system or component’s design or implementation is difficult to understand and verify. In other words, complexity is a function of the internal logic: the number and intricacy of the interfaces and the understandability of the architecture and code. [1] No change required.

### Ratings

[1] No change required.

### Score

I place the mobile game at 18 on the D scale. With a mobile game there will be some working with the OS’s API, but it will be very minimal. The complexity of a game is mostly graphical, which, while complicated, is not nearly as bad as things with a higher complexity level.

## DISP

### Define

Special display requirements, ranging from no UI to very complicated UI.

### Ratings

The reading broke this into 4 categories. I think the 4 category approach is valid, but tweaking is needed to the fields.

1. Simple - Command line or Text only interface.
2. GUI - Requires the use of a stylesheet and a UI system like HTML5/CSS or JavaFX. Simple interaction with the user (e.g., A webpage or simple program like a budget calculator)
3. GUI with more complicated UI or with more difficult interaction with the user (e.g., a 2D game or the creation of a text editor like vim or Word
4. 3D Graphics engine

### Score

Our mobile game will have a 3D graphics engine, thus we give it a score of 4.

## HOST

I am removing this factor. Almost all software is developed on hardware similar to the host or is run/tested against a virtualization of the host. Almost all software is developed in the language it will be deployed in. With these points being the major reason for a HOST variable, I consider HOST no longer relevant to modern development.

## MEMC

### Define

The target system Memory Constraint (MEMC) rating evaluates the development impact of anticipated effort to reduce application memory requirements [1]. No change required.

### Ratings

1.0 - No memory constraints.

1.1 - The application must use less than 1TB memory

1.2 - The application must use less than 10GB memory

1.3 - The application must use less than 100MB memory

1.4 - The application must use less than 1MB memory

### Score

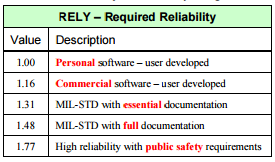
1.25. Mobile development requires extensive focus on memory management. Especially when creating a mobile game. A console or pc game can have gigabytes of information in the form of models and textures that need to be in memory. To create a high quality mobile game requires you to focus on reducing your memory footprint as much as possible in order to provide the level of detail required in today’s gaming environment. The average phone has between 512MB and 2GB Memory. Mobile Gaming is generally geared towards newer hardware, so 1.25 seems like a reasonable score.

## RELY

### Define

The Required Reliability (RELY) parameter evaluates the impact of reliability requirements on the ultimate software development effort. The reliability impact includes the specification or documentation level required by the product development, the level of quality assurance (QA) imposed on the development, and the test and rigor required in the unit and formal qualification testing phases. Also included in RTIM is application availability. If the application requires a particular percentage of uptime (like most web hosts/web programs) or experiences heavy load at certain times of the day, this needs to be factored in as well.

### Ratings

[1] Additionally, each of the five categories increases from none to minor to moderate to large to extreme with regards to uptime requirements and successful continued operation under peak loads.

### Score

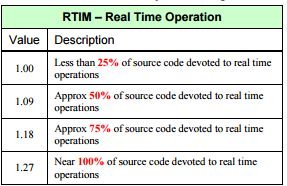
1.16. Apps do not have uptime or peak times, and thus we will focus on the metrics inside the table. A mobile game does not need military standard reliability. But in order for it to be successful it needs to be held to a commercial standard of reliability. Games with excess bugs do not sell well.

## RTIM

### Define

The Real-Time Operation (RTIM) rating evaluates the impact of the fraction of the software product that interacts with the outside environment. [1] No change required

### Ratings

[1] No change required

### Score

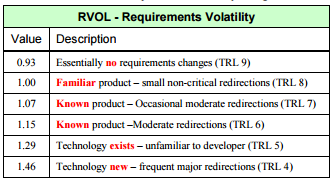
1.09. The game logic for handling real time user input is a significant portion of the programming required for the game. While this handling is less than half of the game, the code the handler calls needs to be very performant in order to give the user the appropriate sense of control over their character. This is especially true in a mobile environment where user input frequently feels “clunky”. Because this notion affects how much of the code is designed, I chose 1.09 over 1.00

## RVOL

### Define

The RVOL rating evaluates the cost penalty due to expected frequency and scope of requirements changes after the baseline Software Requirements Review, and projects the impact of those changes on delivery cost and schedule. [1] In addition to Technology Readiness Levels, some projects require a large amount of flexibility with regard to requirements.

### Ratings

[1] 1.29 and 1.46 can also be reached via large and extreme (respectively) amounts of redirection or requirements change.

### Score

1.29. Many games today are developed to an alpha stage and then given a limited release to receive feedback and course correct the game to match users’ expectations. This can lead to a large amount of requirements volatility as users in these situations have difficult and frequently changing expectations.

## SERC

### Define

The level of protection given to personally identifiable information or otherwise sensitive or classified information. This includes the degree of application for encryption, input sanitation, and other important defensive programming techniques.

### Ratings

1.00 - No sensitive information, no security requirements.

1.25 - Minor levels of personally identifiable information, like email addresses.

1.50 - Moderate levels of personally identifiable information, like full names and addresses.

1.75 - High levels of personally identifiable information, like social security numbers or medical information.

2.00 - Commercially sensitive information (Credit card, Bank Account, Business accounting information, trade secrets, etc..)

2.25 - Government classified information

### Score

1.25 - Since the commercial aspects of the app are handled through the App store, only minor levels of personally identifiable information will required for things like leaderboards or emails about updates or expansions to the game.

# E-Commerce

**Please see the Mobile Game section for information on definitions and ratings. Only scores will be provided for the E-Commerce and Life-Support sections.**

## D

12. Handling money requires shipping estimation, taxes where applicable, processing payment methods, inventory tracking, and a host of other complex interactions.

## DISP

2.5. The GUI, while not terribly complex, is somewhere between a 2D game and a simple GUI for a static webpage or simple program.

## HOST

Eliminated. See rationale in Mobile Game section

## MEMC

1.0. No real memory concerns. Storage for inventory tracking/pricing and sales information may be a long term issue, but isn’t a strong enough concern to increase the value of this variable.

## RELY

1.48. A large effort will be put into both uptime and the ability to handle large loads of users during peak times (like black friday sales).

## RTIM

1.00. There is very little real time code here. You need to make sure you still have inventory to sell, but there is definitely less than 25% of the code dedicated to handling real time events.

## RVOL

1.15. Moderate redirection will occur as the user changes their inventory or wishes to try a new marketing technique that requires changes to your project. This is a known product, so the impacts will be moderate, but not overly high.

## SERC

2.0. This application will handle sensitive financial information for potentially thousands of customers.

# Life Support

## D

4. While not using microcode, this is an extremely complex system due to the interactions between many different pieces of hardware and the extreme cost of failure.

## DISP

1-2. Depending on the design, you could provide text output for diagnostics, or a simple GUI for diagnostics. Most of the output of this machine isn’t on a typical display. It will be the numbers/lights on the equipment.

## HOST

This variable was removed. See Mobile Game section for details.

## MEMC

1.15. I have an extremely difficult time seeing this software being constrained to less than several gigabytes of memory. This would need to be adjusted if the life support system is not running on a typical computer system. (If this is an embedded system in a hospital equipment rack, a much higher MEMC may be required)

## RELY

1.8. It is difficult to make a list of programs that require more reliability than a life support system. 100% reliability while in use will be required. A life is literally on the line.

## RTIM

1.27. The software needs to be able to immediately respond to changing inputs on any of the sensors. Everything is time critical.

## RVOL

.93. Requirements will need to be locked down very early in the development process. With such critical software, extreme methods, like cleanroom software engineering, are likely to be used. Changes to the sensors or requirements are likely to be high enough cost as to scrap the project.

## SERC

1.0. While this system handles medical information, it is not personally identifiable. The output of this program is likely being stored via another system, but this system need not be concerned with the protection of sensitive information.

# References

[1] Software Technology Support Center Cost Analysis Group; "**Section 10 Product Characteristics Evaluation**: Software Development Cost Estimating Guidebook"; Oct 2010;

[http://www.stsc.hill.af.mil/consulting/sw\_estimation/softwareguidebook2010.pdf](https://content.byui.edu/items/8bcbc45e-012a-48e0-800d-7082dd962f15/1/?.vi=file&attachment.uuid=2c5018f3-422e-45eb-a444-67136a1c8b5e)