

Price S Parameter Justification

C Uncompressed:

Item Descriptors

Platform(1.00) - Production Center Contracted Software operating environment

Mgmt Complex(1 00) - Not a multi-national product. Not more than one location working on the development at a single point in time.

External Intergration(0.30) - Based on the concept of Object-Orientation and its class structuring, the CSCIs are much easier to integrate than standard top-down level coding.

Developed CSCI Components 1/3/5 titled: GUI (Graphical User Interface)/TUTOR DEVELOPMENT/SIMULATION

Internal & External Integration(0.30/0.30/0.30) - Based on the concept of Object-Orientation and its inherent class structuring, the CSCI & CSC integration will be greatly simplified versus a standard top-down level coding integration.

Utilization Fraction(0.40/0.40/0.40) - The design of the MST does not require critical timing and memory requirements. Based on the use of an object-oriented language, management of memory utilization is automatically controlled within the language. This reduces the complexity of allocating memory therefore reducing the utilization fraction

Language Descriptors

Language - C Language - C language supports the implementation of object-orientation and is similar to C++ in terms of complexity and functionality. Although Smalltalk may be the primary language for the MST, efficiency of using Smalltalk vs. C is even greater. It provides a reasonable approximation for an object-oriented programming language.

Complexity 1(0.90/0.90/0.90) - Normal experience crew/personnel. Product familiarity. Nominal software tools and Smalltalk experience. Relatively stable requirements.

Source Code(323/30,600/18,500) - # of lines of unique developed GUI/TUTOR/SIMULATION code.

Non-executable SLOC(0.20/0.20/0.20) - standard 20% documentation and comments contained within code.

Complexity2 (1.00/1.00/1.00) - No parallel hardware development and no complicating factors in the hardware deployed as part of the MST system.

Productivity Factor(8.00/6.50/6.50) - The PROFAC given to GUI development is based on the use of multiple automatic code generators for GUI support that are available in industry to date and the increased throughput of code developed from skilled and experience personnel. The PROFAC of 6.5 given to the Tutor development and the Simulation portions of the developed code are based on: 1) Tutor Complexity (i.e. the Application values); 2) Amount of Effort required to produce tutor-specific code; 3) The Programming Language (Smalltalk); and 4) the Tutor platform/hardware.

Identifying characteristics for each of the following application areas within the CSCs as well as justification for our design and code values is shown below:

Data S/R (Data Storage and Retrieval) -- Operation of data storage devices. Database management. Secondary storage handling. Data blocking and deblocking. Hashing techniques. Hardware oriented.

Mix (.10/.30/.15) - 10% of the GUI CSC code, 30% of the TUTOR development CSC code, and 15% of the SIMULATION CSC code will be of this type.

New Design (.50/.50/.50) - Many of these operations are in the form of standard protocols and proven designs which will require less new design.

New Code (.20/.20/.20) - Despite only moderate design effort, the use of tools and development packages will help to mitigate the amount of new code required for these types of operations.

Online Communications -- Machine to machine communications with queing allowed. Timing restrictions not as restrictive as with real time command and control.

Mix (.20/.15/.25) - 20% of the GUI CSC code, 15% of the TUTOR development CSC code, and 25% of the SIMULATION CSC code will be of this type.

New Design (.60/.80/.50) - Many online communications packages exist which will lessen the likelihood that a person coding will have never done so before; in other words, this type of operation is relatively common and its complexity minimal. This type of code will require less upfront design.

New Code (.20/.20/.20) - Many protocols, COTS, and much non-developmental code exists in this area and the probability that a software engineer will have already written similar code before is very high. A large code savings should be realized in this area.

Realtime Command and Control (C&C) -- Machine to machine communications under tight timing constraints. Queing not practicable. Heavy hardware interface. Strict protocol requirements.

Mix (.00/.10/.15) - 0% of the GUI CSC code, 10% of the TUTOR development CSC code, and 15% of the SIMULATION CSC code will be of this type.

New Design (.00/.80/1.00) - Realtime Command & Control requires relatively complex algorithms and there is the least experience in this code area out in the commercial environment. We have decided to invest a substantial amount of time into the design phase with Realtime C & C.

New Code (.00/.20/.20) - Higher design effort should require less actual coding.

Interactive -- Real time man/machine interfaces. Human engineering considerations and error protection very important.

Mix (.05/.00/.00) - 5% of the GUI CSC code and 0% of the TUTOR development CSC and SIMULATION CSC code will be of this type.

New Design (1.00/.00/.00) - A lot of up front design and consideration will go into the interactive portion of the code.

New Code (.20/.00/.00) - Higher design effort will require less actual coding.

Mathematical -- Routine mathematical applications with overriding constraints.

Mix (.30/.35/.35) - 30% of the GUI CSC code, 35% of the TUTOR development CSC code, and 35% of the SIMULATION CSC code will be of this type.

New Design (.80/1.00/1.00) - Relatively simple number-crunching with moderate design implications.

New Code (.20/.20/.20) - Coding of a low complexity and a large upfront design effort should help to realize lower new code percentages.

String Manipulation -- Routine applications with no overriding constraints. Not oriented toward mathematics. Typified by language compilers, sorting, formatting, buffer manipulation, etc.
Mix (.30/.10/.10) - 30% of the GUI CSC code, 10% of the TUTOR development CSC code, and 10% of the SIMULATION CSC code will be of this type.
New Design (.80/1.00/1.00) - Not too complicated; however, a better design will lead to less coding down the road.
New Code (.20/.20/.20) - Very little new code will be required in this code area where higher efficiencies can be realized due to the lower complexity of the code.

Operating Systems -- Task management. Memory management. Heavy hardware interactions. High reliability and strict timing requirements.
Mix (.05/.00/.00) - 5% of the GUI CSC code and 0% of the TUTOR development CSC and SIMULATION CSC code will be of this type.
New Design (1.00/.00/.00) - Relatively complex interactions requiring careful design.
New Code (.20/.00/.00) - Code of this type exists in many forms but with the large amount of design effort and a moderate market of code already existing, the percentage of new code should stay low.

Purchased Components 2/4/6 titled: GUI (COTS), TUTOR (COTS), SIMULATION (COTS)

Language - C Language - Most likely language of the code that the COTS tools will be used to generate will be C or C++.

Source Code(24000/10200/18500) - # of lines of purchased GUI/TUTOR/SIMULATION code.

Non-executable SLOC(0.20/0.20/0.20) - standard 20% documentation and comments contained within code.

Application(3.0/3.5/4.0) - Very close to the mix value arrived at as a result of the inputs above in "C language" which is an accurate representation of the type and amount of work involved, in the area of integration, in particular.

External Integration(0.50/0.50/0.50) - Normal integration. Nominal value for all purchased COTS.

Purchased Cost(\$25K/\$50K/\$50K) - Cost of purchased GUI COTS/TUTOR COTS/SIMULATION COTS

Blue Book Assumptions

Item Descriptors

Platform(1.00) - Production Center Contracted Software operating environment

Mgmt Complex(1.00) - Not a multi-national product. Not more than one location working on the development at a single point in time.

External Intergration(0.30) - Based on the concept of Object-Orientation, the CSCI(Objects) are much easier to integrate than standard top-down level coding.

Component 1 titled: GUI (Graphical User Interface)

Internal & External Integration(0.30) - Based on the concept of Object-Orientation, the CSCI & CSC(Objects) are much easier to integrate than standard top-down level coding.

Utilization Fraction(0.40) - The design of the MST does not require critical timing and memory requirements. Based on the use of an object-oriented language, management of memory utilization is automatically controlled within the language. This reduces the complexity of allocating memory therefore reducing the utilization fraction.

Language Descriptors

Language - C Language - C language supports the implementation of object-orientation and is similar to C++ in terms of complexity and functionality. Although Smalltalk may be the primary language for the MST, efficiency of using Smalltalk vs. C is even greater.

Complexity 1(0.90) - Normal experience crew/personnel. Product familiarity. Nominal software tools and Smalltalk experience and relatively stable requirements.

Source Code(323) - # of lines of unique developed GUI code.

Non-executable SLOC(0.20) - standard 20% documentation and comments contained within code.

Complexity 2(1.00) - No parallel hardware development and no complicating factors in the hardware deployed as part of the MST system.

Productivity Factor(8.00) - The PROFAC given to GUI development is based on the use of multiple automatic code generators for GUI support that are available in industry to date and the increased throughput of code developed from skilled and experience personnel.

Application(3.69) -

New Design(0.76) -

New Code(0.20) -

Data S/R

New Design -

New Code -

Online Communications

New Design -

New Code -

Realtime C&C

New Design -

New Code -

Interactive

New Design -

New Code -

Mathematical

New Design -

New Code -

String Manipulation

New Design -

New Code -

Operating Systems

New Design -

New Code -

Purchased Component 2 titled: GUI (COTS)

Language - C Language - C language supports the implementation of object-orientation and is similar to C++ in terms of complexity and functionality. Although Smalltalk may be the primary language for the MST, efficiency of using Smalltalk vs. C is even greater.

Source Code(323) - # of lines of unique developed GUI code.

Non-executable SLOC(0.20) - standard 20% documentation and comments contained within code.

Application(3.0) -

External Integration(0.50) -

Purchased Cost(\$20K) -