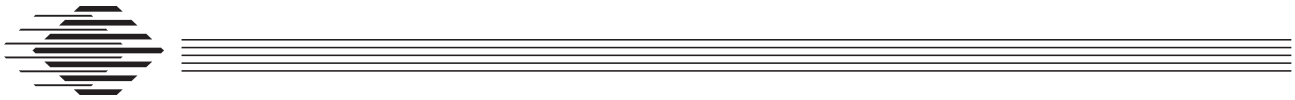
**Technical Report**
CMU/SEI-93-TR-6
ESC-TR-93-183
June 1993

# Taxonomy-Based Risk Identification



Marvin J. Carr
Suresh L. Konda
Ira Monarch
F. Carol Ulrich

Taxonomy Project



**Software Engineering Institute**
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

# Appendix B Taxonomy-Based Questionnaire

# A. Product Engineering

## 1. Requirements

### a. Stability

*[Are requirements changing even as the product is being produced?]*

[1] Are the requirements stable?

(No) (1.a) What is the effect on the system?

- Quality
- Functionality
- Schedule
- Integration
- Design
- Testing

[2] Are the external interfaces changing?

### b. Completeness

*[Are requirements missing or incompletely specified?]*

[3] Are there any TBDs in the specifications?

[4] Are there requirements you know should be in the specification but aren't?

(Yes) (4.a) Will you be able to get these requirements into the system?

[5] Does the customer have unwritten requirements/expectations?

(Yes) (5.a) Is there a way to capture these requirements?

[6] Are the external interfaces completely defined?

### c. Clarity

*[Are requirements unclear or in need of interpretation?]*

[7] Are you able to understand the requirements as written?

(No) (7.a) Are the ambiguities being resolved satisfactorily?

(Yes) (7.b) There are no ambiguities or problems of interpretation?

### d. Validity

*[Will the requirements lead to the product the customer has in mind?]*

[8] Are there any requirements that may not specify what the customer really wants?

(Yes) (8.a) How are you resolving this?

[9] Do you and the customer understand the same thing by the requirements?

(Yes) (9.a) Is there a process by which to determine this?

[10] How do you validate the requirements?

- Prototyping
- Analysis
- Simulations

e. Feasibility

*[Are requirements infeasible from an analytical point of view?]*

[11] Are there any requirements that are technically difficult to implement?

(Yes) (11.a) What are they?

(Yes) (11.b) Why are they difficult to implement?

(No) (11.c) Were feasibility studies done for these requirements?

(Yes) (11.c.1) How confident are you of the assumptions made in the studies?

f. Precedent

*[Do requirements specify something never done before, or that your company has not done before?]*

[12] Are there any state-of-the-art requirements?

- Technologies
- Methods
- Languages
- Hardware

(No) (12.a) Are any of these new to you?

(Yes) (12.b) Does the program have sufficient knowledge in these areas?

(No) (12.b.1) Is there a plan for acquiring knowledge in these areas?

g. Scale

*[Do requirements specify a product larger, more complex, or requiring a larger organization than in the experience of the company?]*

[13] Is the system size and complexity a concern?

(No) (13.a) Have you done something of this size and complexity before?

[14] Does the size require a larger organization than usual for your company?

## 2. Design

a. Functionality

*[Are there any potential problems in meeting functionality requirements?]*

[15] Are there any specified algorithms that may not satisfy the requirements?

(No) (15.a) Are any of the algorithms or designs marginal with respect to meeting requirements?

[16] How do you determine the feasibility of algorithms and designs?

- Prototyping
- Modeling
- Analysis
- Simulation

## b. Difficulty

*[Will the design and/or implementation be difficult to achieve?]*

[17] Does any of the design depend on unrealistic or optimistic assumptions?

[18] Are there any requirements or functions that are difficult to design?

(No) (18.a) Do you have solutions for all the requirements?

(Yes) (18.b) What are the requirements?

- Why are they difficult?

## c. Interfaces

*[Are the internal interfaces (hardware and software) well defined and controlled?]*

[19] Are the internal interfaces well defined?

- Software-to-software
- Software-to-hardware

[20] Is there a process for defining internal interfaces?

(Yes) (20.a) Is there a change control process for internal interfaces?

[21] Is hardware being developed in parallel with software?

(Yes) (21.a) Are the hardware specifications changing?

(Yes) (21.b) Have all the interfaces to software been defined?

(Yes) (21.c) Will there be engineering design models that can be used to test the software?

## d. Performance

*[Are there stringent response time or throughput requirements?]*

[22] Are there any problems with performance?

- Throughput
- Scheduling asynchronous real-time events
- Real-time response
- Recovery timelines
- Response time
- Database response, contention, or access

[23] Has a performance analysis been done?

(Yes) (23.a) What is your level of confidence in the performance analysis?

(Yes) (23.b) Do you have a model to track performance through design and implementation?

### e. Testability

*[Is the product difficult or impossible to test?]*

[24] Is the software going to be easy to test?

[25] Does the design include features to aid testing?

[26] Do the testers get involved in analyzing requirements?

### f. Hardware Constraints

*[Are there tight constraints on the target hardware?]*

[27] Does the hardware limit your ability to meet any requirements?

- Architecture
- Memory capacity
- Throughput
- Real-time response
- Response time
- Recovery timelines
- Database performance
- Functionality
- Reliability
- Availability

### g. Non-Developmental Software

*[Are there problems with software used in the program but not developed by the program?]*

## *If re-used or re-engineered software exists*

[28] Are you reusing or re-engineering software not developed on the program?

(Yes) (28.a) Do you foresee any problems?

- Documentation
- Performance
- Functionality
- Timely delivery
- Customization

### *If COTS software is being used*

[29] Are there any problems with using COTS (commercial off-the-shelf) software?

- Insufficient documentation to determine interfaces, size, or performance
- Poor performance
- Requires a large share of memory or database storage
- Difficult to interface with application software
- Not thoroughly tested
- Not bug free
- Not maintained adequately
- Slow vendor response

[30] Do you foresee any problem with integrating COTS software updates or revisions?

## 3. Code and Unit Test

### a. Feasibility

*[Is the implementation of the design difficult or impossible?]*

[31] Are any parts of the product implementation not completely defined by the design specification?

[32] Are the selected algorithms and designs easy to implement?

### b. Testing

*[Are the specified level and time for unit testing adequate?]*

[33] Do you begin unit testing before you verify code with respect to the design?

[34] Has sufficient unit testing been specified?

[35] Is there sufficient time to perform all the unit testing you think should be done?

[36] Will compromises be made regarding unit testing if there are schedule problems?

### c. Coding/Implementation

*[Are there any problems with coding and implementation?]*

[37] Are the design specifications in sufficient detail to write the code?

[38] Is the design changing while coding is being done?

[39] Are there system constraints that make the code difficult to write?

- Timing
- Memory
- External storage

[40] Is the language suitable for producing the software on this program?

[41] Are there multiple languages used on the program?

(Yes) (41.a) Is there interface compatibility between the code produced by the different compilers?

[42] Is the development computer the same as the target computer?

(No) (42.a) Are there compiler differences between the two?

### *If developmental hardware is being used*

[43] Are the hardware specifications adequate to code the software?

[44] Are the hardware specifications changing while the code is being written?

## 4. Integration and Test

### a. Environment

*[Is the integration and test environment adequate?]*

[45] Will there be sufficient hardware to do adequate integration and testing?

[46] Is there any problem with developing realistic scenarios and test data to demonstrate any requirements?

- Specified data traffic
- Real-time response
- Asynchronous event handling
- Multi-user interaction

[47] Are you able to verify performance in your facility?

[48] Does hardware and software instrumentation facilitate testing?

(Yes) (48.a) Is it sufficient for all testing?

### b. Product

*[Is the interface definition inadequate, facilities inadequate, time insufficient?]*

[49] Will the target hardware be available when needed?

[50] Have acceptance criteria been agreed to for all requirements?

(Yes) (50.a) Is there a formal agreement?

[51] Are the external interfaces defined, documented, and baselined?

[52] Are there any requirements that will be difficult to test?

[53] Has sufficient product integration been specified?

[54] Has adequate time been allocated for product integration and test?

### *If COTS*

[55] Will vendor data be accepted in verification of requirements allocated to COTS products?

(Yes) (55.a) Is the contract clear on that?

c. System

*[System integration uncoordinated, poor interface definition, or inadequate facilities?]*

[56] Has sufficient system integration been specified?

[57] Has adequate time been allocated for system integration and test?

[58] Are all contractors part of the integration team?

[59] Will the product be integrated into an existing system?

(Yes) (59.a) Is there a parallel cutover period with the existing system?

(No) (59.a.1) How will you guarantee the product will work correctly when integrated?

[60] Will system integration occur on customer site?

## 5. Engineering Specialties

a. Maintainability

*[Will the implementation be difficult to understand or maintain?]*

[61] Does the architecture, design, or code create any maintenance difficulties?

[62] Are the maintenance people involved early in the design?

[63] Is the product documentation adequate for maintenance by an outside organization?

b. Reliability

*[Are the reliability or availability requirements difficult to meet?]*

[64] Are reliability requirements allocated to the software?

[65] Are availability requirements allocated to the software?

(Yes) (65.a) Are recovery timelines any problem?

c. Safety

*[Are the safety requirements infeasible and not demonstrable?]*

[66] Are safety requirements allocated to the software?

(Yes) (66.a) Do you see any difficulty in meeting the safety requirements?

[67] Will it be difficult to verify satisfaction of safety requirements?

d. Security

*[Are the security requirements more stringent than the current state of the practice or program experience?]*

[68] Are there unprecedented or state-of-the-art security requirements?

[69] Is it an Orange Book system?

[70] Have you implemented this level of security before?

e. Human Factors

*[Will the system will be difficult to use because of poor human interface definition?]*

[71] Do you see any difficulty in meeting the Human Factors requirements?

(No) (71.a) How are you ensuring that you will meet the human interface requirements?

## *If prototyping*

(Yes) (71.a.1) Is it a throw-away prototype?

(No) (71.a.1a) Are you doing evolutionary development?

(Yes) (71.a.1a.1) Are you experienced in this type of development?

(Yes) (71.a.1a.2) Are interim versions deliverable?

(Yes) (71.a.1a.3) Does this complicate change control?

f. Specifications

*[Is the documentation adequate to design, implement, and test the system?]*

[72] Is the software requirements specification adequate to design the system?

[73] Are the hardware specifications adequate to design and implement the software?

[74] Are the external interface requirements well specified?

[75] Are the test specifications adequate to fully test the system?

## *If in or past implementation phase*

[76] Are the design specifications adequate to implement the system?

- Internal interfaces

# B. Development Environment

## 1. Development Process

### a. Formality

*[Will the implementation be difficult to understand or maintain?]*

[77] Is there more than one development model being used?

- Spiral
- Waterfall
- Incremental

(Yes) (77.a) Is coordination between them a problem?

[78] Are there formal, controlled plans for all development activities?

- Requirements analysis
- Design
- Code
- Integration and test
- Installation
- Quality assurance
- Configuration management

(Yes) (78.a) Do the plans specify the process well?

(Yes) (78.b) Are developers familiar with the plans?

### b. Suitability

*[Is the process suited to the development model, e.g., spiral, prototyping?]*

[79] Is the development process adequate for this product?

[80] Is the development process supported by a compatible set of procedures, methods, and tools?

### c. Process Control

*[Is the software development process enforced, monitored, and controlled using metrics? Are distributed development sites coordinated?]*

[81] Does everyone follow the development process?

(Yes) (81.a) How is this insured?

[82] Can you measure whether the development process is meeting your productivity and quality goals?

### *If there are distributed development sites*

[83] Is there adequate coordination among distributed development sites?

d. Familiarity

*[Are the project members experienced in use of the process? Is the process understood by all staff members?]*

[84] Are people comfortable with the development process?

e. Product Control

*[Are there mechanisms for controlling changes in the product?]*

[85] Is there a requirements traceability mechanism that tracks requirements from the source specification through test cases?

[86] Is the traceability mechanism used in evaluating requirement change impact analyses?

[87] Is there a formal change control process?

(Yes) (87.a) Does it cover all changes to baselined requirements, design, code, and documentation?

[88] Are changes at any level mapped up to the system level and down through the test level?

[89] Is there adequate analysis when new requirements are added to the system?

[90] Do you have a way to track interfaces?

[91] Are the test plans and procedures updated as part of the change process?

## 2. Development System

a. Capacity

*[Is there sufficient work station processing power, memory, or storage capacity?]*

[92] Are there enough workstations and processing capacity for all staff?

[93] Is there sufficient capacity for overlapping phases, such as coding, integration and test?

b. Suitability

*[Does the development system support all phases, activities, and functions?]*

[94] Does the development system support all aspects of the program?

- Requirements analysis
- Performance analysis
- Design
- Coding
- Test
- Documentation
- Configuration management
- Management tracking
- Requirements traceability

c. Usability

*[How easy is the development system to use?]*

[95] Do people find the development system easy to use?

[96] Is there good documentation of the development system?

d. Familiarity

*[Is there little prior company or project member experience with the development system?]*

[97] Have people used these tools and methods before?

e. Reliability

*[Does the system suffer from software bugs, down-time, insufficient built-in back-up?]*

[98] Is the system considered reliable?

- Compiler
- Development tools
- Hardware

f. System Support

*[Is there timely expert or vendor support for the system?]*

[99] Are the people trained in use of the development tools?

[100] Do you have access to experts in use of the system?

[101] Do the vendors respond to problems rapidly?

g. Deliverability

*[Are the definition and acceptance requirements defined for delivering the development system to the customer not budgeted? HINT: If the participants are confused about this, it is probably not an issue from a risk perspective.]*

[102] Are you delivering the development system to the customer?

(Yes) (102.a) Have adequate budget, schedule, and resources been allocated for this deliverable?

## 3. Management Process

a. Planning

*[Is the planning timely, technical leads included, contingency planning done?]*

[103] Is the program managed according to the plan?

(Yes) (103.a) Do people routinely get pulled away to fight fires?

[104] Is re-planning done when disruptions occur?

[105] Are people at all levels included in planning their own work?

[106] Are there contingency plans for known risks?

(Yes) (106.a) How do you determine when to activate the contingencies?

[107] Are long-term issues being adequately addressed?

### b. Project Organization

*[Are the roles and reporting relationships clear?]*

[108] Is the program organization effective?

[109] Do people understand their own and others' roles in the program?

[110] Do people know who has authority for what?

### c. Management Experience

*[Are the managers experienced in software development, software management, the application domain, the development process, or on large programs?]*

[111] Does the program have experienced managers?

- Software management
- Hands-on software development
- With this development process
- In the application domain
- Program size or complexity

### d. Program Interfaces

*[Is there poor interface with customer, other contractors, senior and/or peer managers?]*

[112] Does management communicate problems up and down the line?

[113] Are conflicts with the customer documented and resolved in a timely manner?

[114] Does management involve appropriate program members in meetings with the customer?

- Technical leaders
- Developers
- Analysts

[115] Does management work to ensure that all customer factions are represented in decisions regarding functionality and operation?

[116] Is it good politics to present an optimistic picture to the customer or senior management?

## 4. Management Methods

### a. Monitoring

*[Are management metrics defined and development progress tracked?]*

[117] Are there periodic structured status reports?

(Yes) (117.a) Do people get a response to their status reports?

[118] Does appropriate information get reported to the right organizational levels?

[119] Do you track progress versus plan?

(Yes) (119.a) Does management have a clear picture of what is going on?

### b. Personnel Management

*[Are project personnel trained and used appropriately?]*

[120] Do people get trained in skills required for this program?

(Yes) (120.a) Is this part of the program plan?

[121] Do people get assigned to the program who do not match the experience profile for your work area?

[122] Is it easy for program members to get management action?

[123] Are program members at all levels aware of their status versus plan?

[124] Do people feel it's important to keep to the plan?

[125] Does management consult with people before making decisions that affect their work?

[126] Does program management involve appropriate program members in meetings with the customer?

- Technical leaders
- Developers
- Analysts

### c. Quality Assurance

*[Are there adequate procedures and resources to assure product quality?]*

[127] Is the software quality assurance function adequately staffed on this program?

[128] Do you have defined mechanisms for assuring quality?

(Yes) (128.a) Do all areas and phases have quality procedures?

(Yes) (128.b) Are people used to working with these procedures?

d. Configuration Management

*[Are the change procedures or version control, including installation site(s), adequate?]*

[129] Do you have an adequate configuration management system?

[130] Is the configuration management function adequately staffed?

[131] Is coordination required with an installed system?

(Yes) (131.a) Is there adequate configuration management of the installed system?

(Yes) (131.b) Does the configuration management system synchronize your work with site changes?

[132] Are you installing in multiple sites?

(Yes) (132.a) Does the configuration management system provide for multiple sites?

## 5. Work Environment

a. Quality Attitude

*[Is there a lack of orientation toward quality work?]*

[133] Are all staff levels oriented toward quality procedures?

[134] Does schedule get in the way of quality?

b. Cooperation

*[Is there a lack of team spirit? Does conflict resolution require management intervention?]*

[135] Do people work cooperatively across functional boundaries?

[136] Do people work effectively toward common goals?

[137] Is management intervention sometimes required to get people working together?

c. Communication

*[Is there poor awareness of mission or goals, poor communication of technical information among peers and managers?]*

[138] Is there good communication among the members of the program?

- Managers
- Technical leaders
- Developers
- Testers
- Configuration management
- Quality assurance

[139] Are the managers receptive to communication from program staff?

(Yes) (139.a) Do you feel free to ask your managers for help?

(Yes) (139.b) Are members of the program able to raise risks without having a solution in hand?

[140] Do the program members get timely notification of events that may affect their work?

(Yes) (140.a) Is this formal or informal?

## d. Morale

*[Is there a non-productive, non-creative atmosphere? Do people feel that there is no recognition or reward for superior work?]*

[141] How is morale on the program?

(No) (141.a) What is the main contributing factor to low morale?

[142] Is there any problem keeping the people you need?

## C. Program Constraints

# 1. Resources

### a. Schedule

*[Is the schedule inadequate or unstable?]*

[143] Has the schedule been stable?

[144] Is the schedule realistic?

(Yes) (144.a) Is the estimation method based on historical data?

(Yes) (144.b) Has the method worked well in the past?

[145] Is there anything for which adequate schedule was not planned?

- Analysis and studies
- QA
- Training
- Maintenance courses and training
- Capital equipment
- Deliverable development system

[146] Are there external dependencies which are likely to impact the schedule?

### b. Staff

*[Is the staff inexperienced, lacking domain knowledge, lacking skills, or understaffed?]*

[147] Are there any areas in which the required technical skills are lacking?

- Software engineering and requirements analysis method
- Algorithm expertise
- Design and design methods
- Programming languages
- Integration and test methods
- Reliability
- Maintainability
- Availability
- Human factors
- Configuration management
- Quality assurance
- Target environment
- Level of security
- COTS
- Reuse software
- Operating system
- Database

- Application domain
- Performance analysis
- Time-critical applications

[148] Do you have adequate personnel to staff the program?

[149] Is the staffing stable?

[150] Do you have access to the right people when you need them?

[151] Have the program members implemented systems of this type?

[152] Is the program reliant on a few key people?

[153] Is there any problem with getting cleared people?

### c. Budget

*[Is the funding insufficient or unstable?]*

[154] Is the budget stable?

[155] Is the budget based on a realistic estimate?

(Yes) (155.a) Is the estimation method based on historical data?

(Yes) (155.b) Has the method worked well in the past?

[156] Have features or functions been deleted as part of a design-to-cost effort?

[157] Is there anything for which adequate budget was not allocated?

- Analysis and studies
- QA
- Training
- Maintenance courses
- Capital equipment
- Deliverable development system

[158] Do budget changes accompany requirement changes?

(Yes) (158.a) Is this a standard part of the change control process?

### d. Facilities

*[Are the facilities adequate for building and delivering the product?]*

[159] Are the development facilities adequate?

[160] Is the integration environment adequate?

## 2. Contract

### a. Type of Contract

*[Is the contract type a source of risk to the program?]*

[161] What type of contract do you have? (Cost plus award fee, fixed price,....)

(161a) Does this present any problems?

[162] Is the contract burdensome in any aspect of the program?

- SOW (Statement of Work)
- Specifications
- DIDs (Data Item Descriptions)
- Contract parts
- Excessive customer involvement

[163] Is the required documentation burdensome?

- Excessive amount
- Picky customer
- Long approval cycle

### b. Restrictions

*[Does the contract cause any restrictions?]*

[164] Are there problems with data rights?

- COTS software
- Developmental software
- Non-developmental items

### c. Dependencies

*[Does the program have any dependencies on outside products or services?]*

[165] Are there dependencies on external products or services that may affect the product, budget, or schedule?

- Associate contractors
- Prime contractor
- Subcontractors
- Vendors or suppliers
- Customer furnished equipment or software

## 3. Program Interfaces

### a. Customer

*[Are there any customer problems such as: lengthy document-approval cycle, poor communication, and inadequate domain expertise?]*

[166] Is the customer approval cycle timely?

- Documentation
- Program reviews
- Formal reviews

[167] Do you ever proceed before receiving customer approval?

[168] Does the customer understand the technical aspects of the system?

[169] Does the customer understand software?

[170] Does the customer interfere with process or people?

[171] Does management work with the customer to reach mutually agreeable decisions in a timely manner?

- Requirements understanding
- Test criteria
- Schedule adjustments
- Interfaces

[172] How effective are your mechanisms for reaching agreements with the customer?

- Working groups (contractual?)
- Technical interchange meetings (contractual?)

[173] Are all customer factions involved in reaching agreements?

(Yes) (173.a) Is it a formally defined process?

[174] Does management present a realistic or optimistic picture to the customer?

### *If there are associate contractors*

### b. Associate Contractors

*[Are there any problems with associate contractors such as inadequately defined or unstable interfaces, poor communication, or lack of cooperation?]*

[175] Are the external interfaces changing without adequate notification, coordination, or formal change procedures?

[176] Is there an adequate transition plan?

(Yes) (176.a) Is it supported by all contractors and site personnel?

[177] Is there any problem with getting schedules or interface data from associate contractors?

(No) (177.a) Are they accurate?

## *If there are subcontractors*

c. Subcontractors

*[Is the program dependent on subcontractors for any critical areas?]*

[178] Are there any ambiguities in subcontractor task definitions?

[179] Is the subcontractor reporting and monitoring procedure different from the program's reporting requirements?

[180] Is subcontractor administration and technical management done by a separate organization?

[181] Are you highly dependent on subcontractor expertise in any areas?

[182] Is subcontractor knowledge being transferred to the company?

[183] Is there any problem with getting schedules or interface data from subcontractors?

## *If program is a subcontract*

d. Prime Contractor

*[Is the program facing difficulties with its Prime contractor?]*

[184] Are your task definitions from the Prime ambiguous?

[185] Do you interface with two separate prime organizations for administration and technical management?

[186] Are you highly dependent on the Prime for expertise in any areas?

[187] Is there any problem with getting schedules or interface data from the Prime?

e. Corporate Management

*[Is there a lack of support or micro management from upper management?]*

[188] Does program management communicate problems to senior management?

(Yes) (188.a) Does this seem to be effective?

[189] Does corporate management give you timely support in solving your problems?

[190] Does corporate management tend to micro-manage?

[191] Does management present a realistic or optimistic picture to senior management?

f. Vendors

*[Are vendors responsive to programs needs?]*

[192] Are you relying on vendors for deliveries of critical components?

- Compilers
- Hardware
- COTS

g. Politics

*[Are politics causing a problem for the program?]*

[193] Are politics affecting the program?

- Company
- Customer
- Associate contractors
- Subcontractors

[194] Are politics affecting technical decisions?