Lists of risks

Product size risks

- Estimated size of the product in LOC or FP?
- Degree of confidence in estimated size estimate?
- Estimated size of product in number of programs, files, transactions?
- Percentage deviation in size of product from average for previous products?
- Size of database created or used by the product?
- Number of users of the product?
- Number of projected changes to the requirements for the product? Before delivery? After delivery?
- Amount of reused software?

Business impact risks

- Affect of this product on company revenue?
- Visibility of this product by senior management?
- Reasonableness of delivery deadlines?
- Number of customers who will use this product and the consistency of their needs relative to the product?
- Number of other products/systems with which this product must be interoperable?
- Sophistication of end users?
- Amount and quality of product documentation that must be produced and delivered to the customer?
- Governmental constraints on the construction of the product?
- Costs associated with late delivery?
- Costs associated with a defective product?

Customer related risks

- Have you worked with he customer in the past?
- Does the customer have a solid idea of what is required?
- Has the customer taken the time to write this down?
- Will the customer agree to spend time in formal requirements gathering meetings to identify project scope?
- Is the customer willing to establish rapid communication links with the developer?
- Is the customer willing to participate in reviews?
- Is the customer technically sophisticated in the product area?
- Is the customer willing to let your people do their job that is, will the customer resists looking over your shoulder during technically detailed work?
- Does the customer understand the software engineering process?

Developement environment risks

- Is a software project management tool available?
- Is a software process management tool available?
- Are tools for analysis and design available?
- Do analysis and design tools deliver methods that are appropriate for the product to be built?
- Are compilers or code generators available and appropriate for the product to be built?
- Are testing tools available and appropriate for the product to be built?

- Are software configuration management tools available?
- Does the environment make use of a database or repository?
- Are all the software tools integrated with one another?
- Have members of the project teams received training in each of the tools?
- Are local experts available to answer questions about the tools?
- Is on-line help and documentation for the tools adequate?

Process issue risks

- Does your senior management support a written policy statement that emphasizes the importance of a standard process for software development?
- Has your organization developed a written description of the software process to be used on this project?
- Are staff members signed-up to the software process as it is documented and willing to use it?
- Is the software process used for other projects?
- Has your organization developed or acquired a series of software engineering training courses for managers and technical staff?
- Are published software engineering standards provided for every software developer and software manager?
- Have document outlines and examples been developed for all deliverables defined as part of the software process?
- Are formal technical reviews of the requirements specification, design, and code conducted regularly?
- Are formal technical reviews of test procedures and test cases conducted regularly?
- Are the results of each formal technical review documented, including defects found and resources used?
- Is there some mechanism for ensuring that work conducted on a project conforms with software engineering standards?
- Is configuration management used to maintain consistency among system/software requirements, design, code, and test cases
- Is a mechanism used for controlling changes to customer requirements that impact the software?
- Is there a documented statement of work, software requirements specification, and software development plan for each subcontract?
- Is a procedure followed for tracking and reviewing the performance of subcontractors?

Staff size and experience

- Are the best people available?
- Do the people have the right combination of skills?
- Are enough people available?
- Are staff committed for entire duration of the project?
- Will some staff be working only part time on this project?
- Do staff have the right expectations about the job at hand?
- Have staff received necessary training?
- Will turnover among staff be low enough to allow continuity?

Technical issue risks

- Are facilitated application specification techniques used to aid in communication between the customer and developer?
- Are specific methods used for software analysis?
- Do you use a specific method for data and architecture designs?
- Is more then 90% of your code written in a high order language?
- Are specific conventions for code documentation defined and used?
- Do you use a specific method for test case design?

- Are software tools used to support software planning and tracking activities?
- Are configuration management software tools used to control and track change activity throughout the software process?
- Are software tools used to support the software analysis and design process?
- Are tools used to create software prototypes?
- Are software tools used to support the testing process?
- Are software tools used to support the production and management of documentation?
- Are quality metrics collected for all software projects?
- Are productivity metrics collected for all software projects?

Technology risks

- Is the technology to be built new to your company?
- Do the customer requirements demand the creation of new algorithms, input or output technology?
- Does the software interface with new or unproven hardware?
- Does the software to be built interface with a database system whose function and performance have not been proven in this application area?
- Does the software to be built interface with vendor supplied software products that are unproven?
- Is a specialized user interface demanded by product requirements?
- Do requirements for the product demand the creation of program components that are unlike any previously developed by your organization?
- Do requirements demand the use of new analysis, design, or testing methods?
- Do requirements demand the use of unconventional software development methods, such as formal methods, AI-based approaches, artificial neural networks?
- Do requirements put excessive performance constraints on the product?
- Is the customer uncertain that the functionality requested is "do-able"?

Other potential risks

- Schedule, resources, and product definition have all been dictated by the customer or upper management and are not in balance
- Schedule is optimistic, "best case," rather than realistic, "expected case"
- Schedule omits necessary tasks · Schedule was based on the use of specific team members, but those team members were not available
- Cannot build a product of the size specified in the time allocated
- Product is larger than estimated (in lines of code, function points, or percentage of previous project's size)
- Effort is greater than estimated (per line of code, function point, module, etc.)
- Re-estimation in response to schedule slips is overly optimistic or ignores project history
- Excessive schedule pressure reduces productivity
- Target date is moved up with no corresponding adjustment to the product scope or available resources
- A delay in one task causes cascading delays in dependent tasks
- Unfamiliar areas of the product take more time than expected to design and implement Organization and management
- Project lacks an effective top-management sponsor
- Project languishes too long in fuzzy front end
- Layoffs and cutbacks reduce team's capacity
- Management or marketing insists on technical decisions that lengthen the schedule
- Inefficient team structure reduces productivity
- Management review/decision cycle is slower than expected
- Budget cuts upset project plans

- Are the best people available
- Do the people have the right kind of skills
- Are enough people available
- Are staff committed for the duration
- Will some staff be working part-time
- Do staff have the right expectations about the job at hand
- Has the staff received the necessary training
- Will staff turnover be low enough to allow continuity
- Management makes decisions that reduce the development team's motivation
- Non-technical third-party tasks take longer than expected (budget approval, equipment purchase approval, legal reviews, security clearances, etc.)
- Planning is too poor to support the desired development speed
- Project plans are abandoned under pressure, resulting in chaotic, inefficient development
- Management places more emphasis on heroics than accurate status reporting, which undercuts its ability to detect and correct problems Development environment
- Facilities are not available on time
- Facilities are available but inadequate (e.g., no phones, network wiring, furniture, office supplies, etc.)
- Facilities are crowded, noisy, or disruptive
- Development tools are not in place by the desired time
- Development tools do not work as expected; developers need time to create workarounds or to switch to new tools
- Development tools are not chosen based on their technical merits, and do not provide the planned productivity
- Is a software project management tool available
- Is a software process management tool available
- Are tools for analysis and design available
- Do analysis and design tools deliver methods that are appropriate for the project to be built
- Are compilers or code generators available and appropriate for the product to be built
- Are software configuration management tools available
- Does the environment make use of a database or repository
- Are all software tools integrated with each other
- Have members of the project team received training in each of the tools
- Are local experts available to answer questions about the tools
- Is on-line help and documentation for the tools available End users
- End-user insists on new requirements
- End-user ultimately finds product to be unsatisfactory, requiring redesign and rework
- End-user does not buy into the project and consequently does not provide needed support
- End-user input is not solicited, so product ultimately fails to meet user expectations and must be reworked Customer
- Customer insists on new requirements
- Customer review/decision cycles for plans, prototypes, and specifications are slower than expected
- Customer will not participate in review cycles for plans, prototypes, and specifications or is incapable of doing so-resulting in unstable requirements and time-consuming changes
- Customer communication time (e.g., time to answer requirements-clarification questions) is slower than expected
- Customer insists on technical decisions that lengthen the schedule
- Customer micro-manages the development process, resulting in slower progress than planned
- Customer-furnished components are a poor match for the product under development, resulting in extra design and integration work
- Customer-furnished components are poor quality, resulting in extra testing, design, and integration work and in extra customer-relationship management
- Customer-mandated support tools and environments are incompatible, have poor performance, or have inadequate functionality, resulting in reduced productivity

- Customer will not accept the software as delivered even though it meets all specifications
- Customer has expectations for development speed that developers cannot meet
- Have you worked with the customer in the past
- Does the customer have a solid idea of what is required
- Will the customer agree to spend time in formal requirements gathering meetings to identify project scope
- Is the customer willing to participate in reviews
- Is the customer technically sophisticated in the product area
- Is the customer willing to let your people do their job - that is will the customer resist looking over your shoulder and make changes as you go
- Does the customer understand the software process Contractors
- Contractor does not deliver components when promised
- Contractor delivers components of unacceptably low quality, and time must be added to improve quality
- Contractor does not buy into the project and consequently does not provide the level of performance needed
- Requirements have been baselined but continue to change
- Requirements are poorly defined, and further definition expands the scope of the project
- Additional requirements are added
- Vaguely specified areas of the product are more time-consuming than expected Product and product size
- Error-prone modules require more testing, design, and implementation work than expected
- Unacceptably low quality requires more testing, design, and implementation work to correct than expected
- Development of the wrong software functions requires redesign and implementation
- Development of the wrong user interface results in redesign and implementation
- Development of extra software functions that are not required (gold-plating) extends the schedule
- Meeting product's size or speed constraints requires more time than expected, including time for redesign and reimplementation
- Strict requirements for compatibility with existing system require more testing, design, and implementation than expected
- Requirements for interfacing with other systems, other complex systems, or other systems that are not under the team's control result in unforeseen design, implementation, and testing
- Pushing the computer science state-of-the-art in one or more areas lengthens the schedule unpredictably
- Requirement to operate under multiple operating systems takes longer to satisfy than expected
- Operation in an unfamiliar or unproved software environment causes unforeseen problems
- Operation in an unfamiliar or unproved hardware environment causes unforeseen problems
- Development of a kind of component that is brand new to the organization takes longer than expected
- Dependency on a technology that is still under development lengthens the schedule
- Estimated size of the product in LOC or FP
- Degree of confidence in estimated size estimate
- Estimates size of product in number of programs, files, transactions
- Percentage deviation in size of product from average for previous products
- Size of data base created or used by the product · Number of users of the product
- Number of projected changes to the requirements for the product before delivery after delivery
- Amount of reused software Business impact risks
- Effect of this product on company revenue
- Visibility of this product to senior management
- Reasonableness of delivery deadline
- Number of customers who will use this product and the consistency of their needs relative to the product
- Number of other products/systems with which this product must be interoperable
- Sophistication of end users
- Amount and quality of product documentation that must be produced and delivered to the customer
- Governmental constraints on the construction of the product

- Costs associated with late delivery
- Costs associated with a defective product External environment
- Product depends on government regulations, which change unexpectedly
- Product depends on draft technical standards, which change unexpectedly Technology risks
- Is the technology to be built new to your organization
- Do the customers requirements demand the creation of new algorithms or input and output technology
- Does the software with new or unproven hardware
- Does the software interface with vender supplied software
- Does the software interface with unproven database technology
- Is a specialized user interface demanded by product requirements
- Do requirements for the product demand the creation of program components that are unlike any previously developed by your organization
- Do requirements demand the use of new analysis, design, or testing methods
- Do requirements demand the use of unconventional software development methods
- Do requirements put excessive performance constraints on the product
- Is the customer uncertain that the functionality requested is doable Personnel
- Hiring takes longer than expected
- Task prerequisites (e.g., training, completion of other projects, acquisition of work permit) cannot be completed on time
- Poor relationships between developers and management slow decision making and follow through
- Team members do not buy into the project and consequently does not provide the level of performance needed
- Low motivation and morale reduce productivity
- Lack of needed specialization increases defects and rework
- Personnel need extra time to learn unfamiliar software tools or environment
- Personnel need extra time to learn unfamiliar hardware environment
- Personnel need extra time to learn unfamiliar programming language
- Contract personnel leave before project is complete
- Permanent employees leave before project is complete
- New development personnel are added late in the project, and additional training and communications overhead reduces existing team members' effectiveness
- Team members do not work together efficiently
- Conflicts between team members result in poor communication, poor designs, interface errors, and extra rework
- Problem team members are not removed from the team, damaging overall team motivation
- The personnel most qualified to work on the project are not available for the project
- The personnel most qualified to work on the project are available for the project but are not used for political or other reasons
- Personnel with critical skills needed for the project cannot be found
- Key personnel are available only part time
- Not enough personnel are available for the project
- People's assignments do not match their strengths
- Personnel work slower than expected
- Sabotage by project management results in inefficient scheduling and ineffective planning
- Sabotage by technical personnel results in lost work or poor quality and requires rework Design and implementation
- Overly simple design fails to address major issues and leads to redesign and reimplementation
- Overly complicated design requires unnecessary and unproductive implementation overhead
- Inappropriate design leads to redesign and reimplementation
- Use of unfamiliar methodology results in extra training time and in rework to fix first-time misuses of the methodology
- Product is implemented in a low level language (e.g., assembler), and productivity is lower than expected

- Necessary functionality cannot be implemented using the selected code or class libraries; developers must switch to new libraries or custom-build the necessary functionality
- Code or class libraries have poor quality, causing extra testing, defect correction, and rework
- Schedule savings from productivity enhancing tools are overestimated
- Components developed separately cannot be integrated easily, requiring redesign and rework Process
- Amount of paperwork results in slower progress than expected
- Inaccurate progress tracking results in not knowing the project is behind schedule until late in the project
- Upstream quality-assurance activities are shortchanged, resulting in time-consuming rework downstream
- Inaccurate quality tracking results in not knowing about quality problems that affect the schedule until late in the project
- Too little formality (lack of adherence to software policies and standards) results in miscommunications, quality problems, and rework
- Too much formality (bureaucratic adherence to software policies and standards) results in unnecessary, time-consuming overhead
- Management-level progress reporting takes more developer time than expected
- Half-hearted risk management fails to detect major project risks
- Software project risk management takes more time than expected
- Does senior management support a written policy statement that emphasizes the importance of a standard process for software development
- Has your organization developed a written description of the software process to be use on this project
- Are staff members "signed up" to the software process as it is documented and willing to use it
- Is the software process used for other projects
- Has your organization developed or acquired a series of software engineering training courses for managers and technical staff
- Are published software engineering standards provided for every software developer and software manager
- Have documentation outlines and examples been developed for all deliverables defined as part of the software process
- Are formal technical reviews of the requirements specification, design, and code conducted regularly
- Are formal technical reviews of test procedures and test cases conducted regularly
- Are the results of formal reviews documented, including errors found and resources used
- Is there some mechanism for ensuring that work conducted on a project conforms with software engineering standards
- Is configuration management used to maintain consistency among system software requirements, design, code, and test cases
- Is a mechanism used for controlling changes to customer requirements that impact the software
- Is there a documented statement of work, a software requirements specification and a software development for each subcontract
- Is a procedure followed for tracking and reviewing the performance of subcontractors
- Are facilitated application specification techniques used to aid in communication between the customer and developer
- Are specific methods used for software analysis
- Do you use a specific method for data and architectural design
- Is more than 90 percent of your code written in a high order language · Are specific conventions for code documentation defined and used
- Do you use specific methods for test case design
- Are software tools used to support planning and tracking activities
- Are configuration management software tools used to control and track change activity throughout the software process ·
- Are software tools used to support the software analysis and design process

- Are tools used to create software prototypes
- Are software tools used to support the testing process
- Are software tools used to support the production and management of documentation
- Are quality metrics collected for all software projects
- Are productivity metrics collected for all software projects

Note: the main sources of risks came from Pressman 1997 and construx.com