Software engineering

Engineering

Software and Software Engineering Overview Software Is designed and built by software engineers. Software Is used by virtually everyone In society. Software is pervasive in our commerce, our culture, and our everyday lives. Software engineers have a moral obligation to build reliable software that does no harm to other people. Software engineers view computer software, as being made up of the programs, documents, and data required to design and build the system. Software users are only concerned with whether or not software products meet their expectations and make their tasks easier to complete.

Important Questions for Software Engineers Why does It take so long to get software fleshed? Why are development costs so high? Why can't we find all errors before we give the software to our customers? Why do we spend so much time and effort maintaining existing programs? Why do we continue to have difficulty in measuring progress as software is being developed? Software re Software Is both a product and a vehicle for delivering a product (information). Software Is engineered not manufactured. Software does not wear out, but it does deteriorate.

Industry Is moving toward component-based software construction, but most software is still custom-built. Software Application Domains System software Application software Engineering or Scientific Software Embedded software Product-line software (Includes entertainment software) Web-Applications Artificial Intelligence software Open-world computing Creating software to allow machines of all sizes to communicate with each other across vast networks Intercourse Architecture simple and sophisticated applications that benefit targeted end-user markets worldwide Open Source

Distributing source code for computing applications so customers can make local modifications easily and reliably Reasons for Legacy System Evolution Software must be adapted to meet needs of new computing environments or technology Software must be enhanced to implement new business requirements Software must be extended to make it interoperable with more modern system components Software must be re-architect to make it viable within a network environment Unique Nature of Web APS Network intensive Concurrency Unpredictable load Availability (24/7/365) Data driven Content sensitive

Continuous evolution Immediacy (short time to market) Security Aesthetics Software Engineering Realities Problem should be understood before software solution is developed Design is a pivotal activity Software should exhibit high quality Software should be maintainable Software Engineering to obtain reliable and efficient software in an economical manner. Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software. Software engineering encompasses a process, management techniques, technical methods, and the use of lolls.

Generic Software Process Framework Communication (customer collaboration and requirement gathering) Planning (establishes engineering work plan, describes technical risks, lists resources required, work products produced, and defines work schedule) Modeling (creation of models to help developers and customers understand the requires and software design) Construction (code generation and testing) Deployment (software delivered for customer evaluation and feedback) Software Engineering Umbrella

Activities Software project tracking and control (allows team to assess progress and take reactive action to maintain schedule) Risk management (assess risks that may affect project outcomes or quality) Software quality assurance (activities required to maintain software quality) Technical reviews (assess engineering work products to uncover and remove errors before they propagate to next activity) Measurement (define and collect process, project, and product measures to assist team in delivering software meeting customer needs) Software configuration management (manage effects of change) Risibility management (defines criteria for work product reuse and establish mechanisms to achieve component reuse) Work product reparation and production (activities to create models, documents, logs, forms, lists, etc. Attributes for Comparing Process Models Overall flow and level of interdependencies among tasks Degree to which work tasks are defined within each framework activity Degree to which work products are identified and required Manner in which quality assurance activities are applied Manner in which project tracking and control activities are applied Overall degree of detail and rigor of process description Degree to which stakeholders are involved in the project Level of autonomy given to project team Degree to which team organization and roles are prescribed Understand the problem (communication and analysis) Plan a solution (software design) Carry out the plan (code generation) Examine the result for accuracy (testing and quality assurance) Understand the Problem Who are the stakeholders? What functions and features are required to solve the problem? Is it possible to create smaller problems that are easier to understand?

Can a graphic analysis model be created? Plan the Solution Have you seen similar problems before? Has a similar problem been solved? Can readily solvable subprograms be defined? Can a design model be created? Carry Out the Plan Does solution conform to the plan? Is each solution component provably correct? Examine the Result Is it possible to test each component part of the solution? Does the solution produce results that conform to the data, functions, and features required? Software Practice Core Principles 1 . Software exists to provide value to its users 2. Keep it simple stupid (KISS) 3. Clear vision is essential to the success of any software project 4.

Always specify, design, and implement knowing that someone else will have to understand what you have done to carry out his or her tasks 5. Be open to future changes, don't code yourself into a corner 6. Planning ahead for reuse reduces the cost and increases the clear complete thought before any action almost always produces better results Software Creation Almost every software project is precipitated by a business need (e. G. Correct a system defect, adapt system to changing environment, extend existing system, create new system) Many times an engineering effort will only succeed if the software created for the project succeeds The market will only accept a product is the software embedded within it meets the customer's stated or unstated needs