Software development process

Engineering

The Importance of high quality software Software development is a vital activity in modern American society, and is likely to have increasing significance in the future. Software manages our bank accounts, pays our salaries, controls the aircraft we fly in, regulates power generation and distribution, controls our communications, etc.

Characteristics of high quality software High quality software shares the following obvious attributes: high quality software Is intuitive and easy to use the right things happen " automatically" It Is efficient - people use computers to get things done quickly above all, high quality software is it always produces the advertised results and does not crash!

The need for correct precision in the specification of software The notion that software components can be reused is a principal motivation of object-oriented programming, and has virtually become a postulate of programming. To reuse a previously written software component (or create a new one), a software engineer must have a precise description of its behavior. This precision is essential as even a minor misconception of the function of a component that is at he outset may cause serious errors that are difficult and expensive to correct later in the process.

Typical software development phases Software development models commonly subdivide the process into phases similar to the following : requirements analysis: determine user needs : describe precisely what the role of the software will be design: determine how to realize the software, and devise overall organization implementation: formulate the algorithms and program(s) verification: certify that the program(s) meet the

specification maintenance: perform going changes and corrections after the software is in use The role of formal methods Formal methods are Intended to systematize and Introduce rigor Into all the phases of software development.

This helps us to avoid overlooking critical Issues, provides a standard means to record various assumptions and decisions, and forms a basis for consistency among many related activities. By providing precise and unambiguous description mechanisms, formal methods facilitate the understanding required to coalesce the various phases of software development into a successful endeavor. The programming language used for software development furnishes precise syntax and semantics for the Implementation phase, and this has been true since people began writing programs. But precision in all but this one phase of software development must derive from other sources.

The term " formal methods" pertains to a broad collection of formalisms and abstractions intended to support a comparable level of precision for other phases of software development. While this includes issues currently under active development, several methodologies have reached a level of maturity that can be of benefit to practitioners. There Is a discernible tendency to the books by Denver, Ice, and Woodcock & Looms). Many such topics do indeed support software engineering and it is neither possible nor desirable to avoid these topics when pursuing formal methods. But we will not take the approach that applying discrete mathematics to software engineering assures germane formal methods. The overriding concern of software engineering is the creation of high quality software systems.

With " formal methods" we pursue melding those things that nurture rigor and precision into this endeavor. While our focus on the activities that precede the actual programming itself does lead to machine independent abstractions often associated with mathematics, much of the material has been developed (or tailored) to suit the context of software creation. Specific formalisms that will occupy our attention include: algebraic specification (including BOX) used for specification and verification, predicate logic (including Z) used for specification and verification, catechists used for specification of " reactive" systems ML used primarily for design, and also for requirements analysis.