***Requirements Errors***

***Lecture # 14***

***Today’s Topics***

* Requirements errors
* Addressing requirements errors

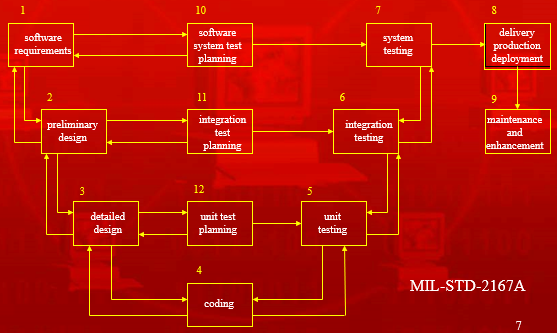
***Requirements Error/Defect***

* A deficiency in the requirements quality that can hamper software development

***Requirements Errors***

* Errors and omissions find their way in different requirements documents
* If not removed, requirements errors usually flow downstream into design, code, and user manuals
* It is difficult to detect requirements errors once they flow downstream
* Requirements errors are most expensive to eliminate

***Software Development Process***



***Types of Requirements Errors***

* ***Errors of omission***
* ***Errors of commission***
* ***Errors of clarity and ambiguity***
* ***Errors of speed and capacity***

***Errors of Omission***

* Errors of omission are most common among requirements errors
* Domain experts easily forget to convey domain knowledge to requirements engineers, because they consider that to be obvious and implicit

***Errors of Clarity and Ambiguity***

* Second most common errors are those of clarity and ambiguity
* Primarily, because natural languages (like English) are used to state requirements, while such languages are themselves ambiguous
* For example: object

***Errors of Commission***

* Errors of commission can also find their way into the requirements documents

***Performance Errors***

* Performance, that is errors of speed and capacity, are also found in requirements
* Primarily, these occur due to conflicting understanding or competing needs of different stakeholders

***Negative Impact of Requirements Errors***

* The resulting software may not satisfy user’s real needs
* Multiple interpretations of requirements may cause disagreements between customers and developers, wasting time and money, and perhaps resulting in lawsuits
* Negative impact on humans
  + Unsatisfied customers and developers
  + Lack of interest in automation of processes
  + Blame game

***Addressing Requirements Errors***

* ***Prevention***
* ***Removal***

***Prevention vs. Removal***

* For requirements errors, prevention is usually more effective than removal
* Joint application development (JAD), quality function deployment (QFD), and prototyping are more effective in defect prevention
* Requirements inspections and prototyping play an important role in defect removal

***Defect Prevention***

* Don’t let defects/errors become part of the requirements document or requirements model in the first place
* How is it possible?
* Understanding application domain and business area is the first step in defect prevention
* Training in different requirements engineering activities (elicitation, analysis and negotiation, specification, and validation) is also very important for defect prevention
* Allocating enough time to conduct requirements engineering activities also is very important in this regard
* Willing and active participation of stakeholders in different activities of requirements engineering. That is why JAD is very useful in defect prevention as far as requirements errors are concerned
* An overall commitment to quality and emphasis on using documented processes is also a very important
* An overall commitment to process improvement

***Summary***

* Introduced the concept of requirements errors and types of requirements errors
* Discussed the impact of requirements errors
* Discussed error prevention in requirements