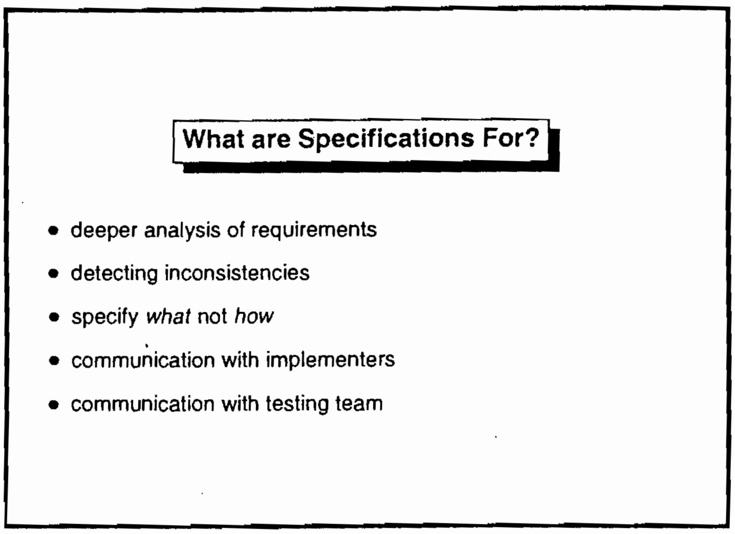
p2q1c
LCP

Slide 402



What are Specifications For?

- deeper analysis of requirements
- detecting inconsistencies
- specify *what* not *how*
- communication with implementers
- communication with testing team

A formal specification is essential if you are going to prove correctness, or to support transformation into correct code. Less ambitiously, formal proof can be used to derive properties from a specification; this could reveal inconsistencies early. The specification is also useful in itself. Studies have shown that attempting to write a formal specification stimulates deeper thinking about the requirements, showing up ambiguities hidden in English.

The ConForm Project [6] is investigating the costs and benefits of using formal methods in building a small security-critical system. Two teams are independently developing a so-called trusted gateway. One team is using fairly conventional structured methods; the other augments these methods by writing a formal specification. They are using VDM, the Vienna Development Method, which has many adherents. The project is monitoring the development process, comparing the effort required to complete each phase, the quality of the documents produced, etc.

Early in the project they noticed the team using formal methods asked many more questions concerned with clarifying the requirements. The job of the trusted gateway is to take a stream of messages and forward each message either to a 'secret' or 'non-secret' output port; the decision is based upon certain keywords that may appear in messages.

Messages are limited to 10K. The formal methods team asked whether this limit included the message delimiters (it did). If a message contains both 'secret' and 'non-secret' keywords then it is regarded as secret. However, the formal methods team noticed the possibility that a 'non-secret' keyword could contain a 'secret' keyword as a substring. The developers had to go back to the customers to find out that such occurrences of 'secret' keywords should be ignored.

These are perfect examples of ambiguities that lurk in English descriptions, and that could lead to obscure errors. How many messages will be under 10K if delimiters are ignored, and over 10K if they are counted? The precision of a formal specification will help the implementers build a correct system, particularly if they have tool support. And the specification will help the testing team identify awkward cases to cover in test data.

It's not a bug, it's a feature! — formal specifications can help put an end to this (though it is partly a problem of requirements). Recall our problems in the first lecture.