SE 2AA4, CS 2ME3 (Introduction to Software Development)

Winter 2018

# 22 Intro to Specification Continued (Ch. 5) DRAFT

Dr. Spencer Smith

Faculty of Engineering, McMaster University

December 15, 2017

McMaster University

# 22 Intro to Specification Continued (Ch. 5) DRAFT

- Administrative details
- Questions on midterm?
- Uses of specification
- Qualities of a good specification
- Classification of specification styles
- Examples
- How to verify a specification

# Administrative Details

TBD

# Midterm Examination

TBD

# Uses of Specification Continued

- Requirements for implementation
  - design process is a chain of specification (i.e., definition → implementation → verification steps)
    - requirements specification refers to definition of external behavior - design specification must be verified against it
    - design specification refers to definition of the software architecture -code must be verified against it
- A reference point during maintenance
  - Corrective maintenance only changes the implementation
  - Adaptive and perfective maintenance occur because of requirements changes
    - The requirements specification must change accordingly
  - Specification clarifies whether the change involves a modification to the interface or the implementation - if just the implementation than client modules will be unaffected

# Specification Qualities

- What are the important qualities for a specification?

# Specification Qualities

- The qualities we previously discussed (usability, maintainability, reusability, verifiability etc.)
- Clear, unambiguous, understandable
- Consistent
- Complete
  - Internal completeness
  - External completeness
- Incremental
- Validatable
- Abstract

# Clear, Unambiguous, Understandable

- Specification fragment for a word-processor
  - Selecting is the process of designating areas of the document that you want to work on. Most editing and formatting actions require two steps: first you select what you want to work on, such as text or graphics; then you initiate the appropriate action.
- What are the potential problems with this specification?

# Clear, Unambiguous, Understandable

- Specification fragment for a word-processor
  - Selecting is the process of designating areas of the document that you want to work on. Most editing and formatting actions require two steps: first you select what you want to work on, such as text or graphics; then you initiate the appropriate action.
- What are the potential problems with this specification?
  - Can an area be scattered?
  - Can both text and graphics be selected?

# Clear, Unambiguous, Understandable

- Specification fragment from a real safety-critical system
  - The message must be triplicated. The three copies must be forwarded through three different physical channels. The receiver accepts the message on the basis of a two-out-of-three voting policy.
- What is a potential problems with this specification?

# Clear, Unambiguous, Understandable

- Specification fragment from a real safety-critical system
  - The message must be triplicated. The three copies must be forwarded through three different physical channels. The receiver accepts the message on the basis of a two-out-of-three voting policy.
- What is a potential problems with this specification?
  - Can a message be accepted as soon as we receive 2 out of 3 identical copies, or do we need to wait for receipt of the 3rd

# Unambiguous, Validatable

- Specification fragment for an end-user program
  - The program shall be user friendly.
- What is a potential problems with this specification?

# Unambiguous, Validatable

- Specification fragment for an end-user program
  - The program shall be user friendly.
- What is a potential problems with this specification?
  - What does it mean to be user friendly?
  - Who is a typical user?
  - How would you measure success or failure in meeting this requirement?

# Unambiguous, Validatable

- Specification fragment for a linear solver
  - Given $A$ and $b$, solve the linear system $Ax = b$ for $x$, such that the error in any entry of $x$ is less than 5 %.
- What is a potential problems with this specification?

# Unambiguous, Validatable

- Specification fragment for a linear solver
  - Given $A$ and $b$, solve the linear system $Ax = b$ for $x$, such that the error in any entry of $x$ is less than 5 %.
- What is a potential problems with this specification?
  - Is $A$ constrained to be square?
  - Can $A$ be singular?
  - Even if the problem is made completely unambiguous, the requirement cannot be validated.

# Consistent

- Specification fragment for a word-processor
  - The whole text should be kept in lines of equal length. The length is specified by the user. Unless the user gives an explicit hyphenation command, a carriage return should occur only at the end of a word.
- What is a potential problems with this specification?

# Consistent

- Specification fragment for a word-processor
  - The whole text should be kept in lines of equal length. The length is specified by the user. Unless the user gives an explicit hyphenation command, a carriage return should occur only at the end of a word.
- What is a potential problems with this specification?
  - What if the length of a word exceeds the length of the line?

# Same Symbol/Term Different Meaning

- Can you think of some symbols/terms that have different meanings depending on the context?

# Consistent

- Language and terminology must be consistent within the specification
- Potential problem with homonyms, for instance consider the symbol $\sigma$
  - Represents standard deviation
  - Represents stress
  - Represents the Stefan-Boltzmann constant (for radiative heat transfer)
- Changing the symbol may be necessary for consistency, but it could adversely effect understandability
- Potential problem with synonyms
  - Externally funded graduate students, versus eligible graduate students, versus non-VISA students
  - Material behaviour model versus constitutive equation

# Complete

- Internal completeness
    - The specification must define any new concept or terminology that it uses
        - A glossary is helpful for this purpose
- External completeness
    - The specification must document all the needed requirements
        - Difficulty: when should one stop?

# Incremental

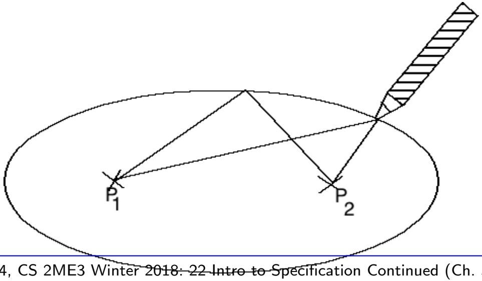
- Referring to the specification process
  - Start from a sketchy document and progressively add details
  - A document template can help with this
- Referring to the specification document
  - Document is structured and can be understood in increments
  - Again a document template can help with this

# Classification of Specification Styles

- Informal, semi-formal, formal
- Operational
  - Behaviour specification in terms of some abstract machine
- Descriptive
  - Behaviour described in terms of properties
- The module state machine specification that we use is a mix of operational and descriptive specification - Why?

# Example Operational Specification

- Specification of a geometric figure $E$
- $E$ can be drawn as follows
  1. Select two points $P_1$ and $P_2$ on a plane
  2. Get a string of a certain length and fix its ends to $P_1$ and $P_2$
  3. Position a pencil as shown in the next figure
  4. Move the pen clockwise, keeping the string tightly stretched, until you reach the point where you started drawing



# Example Descriptive Specification

Geometric figure $E$ is described by the following equation

$$ax^2 + by^2 + c = 0$$

where $a$, $b$ and $c$ are suitable constants

# Another Example

- Operational specification
  - "Let *a* be an array of *n* elements. The result of its sorting is an array *b* of *n* elements such that the first element of *b* is the minimum of *a* (if several elements of a have the same value, any one of them is acceptable); the second element of *b* is the minimum of the array of $n-1$ elements obtained from *a* by removing its minimum element; and so on until all *n* elements of *a* have been removed."
- Descriptive specification
  - "The result of sorting array *a* is an array *b* which is a permutation of *a* and is sorted."
  - How can we further specify (formalize) the notion of sorted?

# Another Example

- Operational specification
  - "Let *a* be an array of *n* elements. The result of its sorting is an array *b* of *n* elements such that the first element of *b* is the minimum of *a* (if several elements of a have the same value, any one of them is acceptable); the second element of *b* is the minimum of the array of $n - 1$ elements obtained from *a* by removing its minimum element; and so on until all *n* elements of *a* have been removed."
- Descriptive specification
  - "The result of sorting array *a* is an array *b* which is a permutation of *a* and is sorted."
  - How can we further specify (formalize) the notion of sorted?
  - $\text{sorted}(A) \equiv \forall(i : \mathbb{N} | 0 \leq i \leq (|A| - 2) : A[i] \leq A[i + 1])$

# Homework Exercise

- Consider the **line formatter** specification and
  1. How well does the specification do with respect to the following qualities: abstract, correct, unambiguous, complete, consistent and verifiable?
  2. For a requirement specification like that given, what are the advantages and disadvantages of maintaining both a formal specification and a natural language specification?
- Even spending 5 minutes thinking about will help when we discuss next week
- In repo
  - The line formatter specification
  - Meyer (1985) "On Formalism in Specification"