SOFTWARE QUALITY FACTORS

CHAPTER 2: UNDERSTANDING THE CRITICAL FACTORS AFFECTING SOFTWARE QUALITY

INTRODUCTION

- SOFTWARE QUALITY FACTORS DEFINE THE ATTRIBUTES CONTRIBUTING TO SOFTWARE EFFECTIVENESS, RELIABILITY, AND MAINTAINABILITY.
- ENSURE SOFTWARE RELIABILITY AND EFFICIENCY.
- IMPROVE USER SATISFACTION.
- REDUCE MAINTENANCE COSTS.

QUALITY FACTORS

- SOFTWARE QUALITY FACTORS DEFINE THE ATTRIBUTES THAT CONTRIBUTE TO THE OVERALL **EFFECTIVENESS, RELIABILITY,** AND **MAINTAINABILITY** OF A SOFTWARE SYSTEM.
- THESE FACTORS DETERMINE **HOW WELL** A SOFTWARE PRODUCT **MEETS** USER EXPECTATIONS, INDUSTRY STANDARDS, AND BUSINESS REQUIREMENTS.
- UNDERSTANDING AND IMPLEMENTING SOFTWARE QUALITY FACTORS IS CRUCIAL FOR ENSURING SOFTWARE THAT IS **FUNCTIONAL, SECURE, AND ADAPTABLE TO FUTURE NEEDS.**

QUALITY FACTORS

- THE GREAT VARIETY OF ISSUES RELATED TO THE VARIOUS ATTRIBUTES OF SOFTWARE AND ITS USE AND MAINTENANCE, AS DEFINED IN SOFTWARE REQUIREMENTS DOCUMENTS, CAN BE CLASSIFIED INTO CONTENT GROUPS CALLED ***QUALITY FACTORS***.

- WE EXPECT THE **TEAM** RESPONSIBLE FOR **DEFINING** THE SOFTWARE REQUIREMENTS OF A SOFTWARE SYSTEM TO **EXAMINE** THE **NEED** TO DEFINE **THE REQUIREMENTS** THAT BELONG TO EACH FACTOR.

- SOFTWARE REQUIREMENT DOCUMENTS ARE EXPECTED TO DIFFER IN THE EMPHASIS PLACED ON THE VARIOUS FACTORS, A REFLECTION OF THE DIFFERENCES TO BE FOUND AMONG SOFTWARE PROJECTS

SIGNIFICANCE OF SOFTWARE QUALITY FACTORS

- ENSURE SOFTWARE RELIABILITY AND EFFICIENCY.
- IMPROVE USER EXPERIENCE AND SATISFACTION.
- REDUCE MAINTENANCE COSTS AND INCREASE SOFTWARE LIFESPAN.
- FACILITATE COMPLIANCE WITH INDUSTRY STANDARDS (E.G., ISO 25010, IEEE 730).
- ENHANCE SOFTWARE SECURITY AND PREVENT SYSTEM FAILURES.
- CONTRIBUTE TO BUSINESS SUCCESS BY ENSURING SOFTWARE MEETS MARKET DEMANDS.
- HELP IN RISK MITIGATION BY IDENTIFYING POTENTIAL DEFECTS EARLY IN THE DEVELOPMENT CYCLE.

RELATIONSHIP BETWEEN SOFTWARE QUALITY FACTORS AND SOFTWARE DEVELOPMENT

- THESE FACTORS SERVE AS BENCHMARKS FOR SOFTWARE ENGINEERS.
- INFLUENCE THE DESIGN, DEVELOPMENT, TESTING, AND MAINTENANCE PHASES.
- TRADE-OFFS MAY EXIST BETWEEN DIFFERENT FACTORS, SUCH AS USABILITY VS. SECURITY…….. WHICH TO CHOOSE ??
- A UNIVERSAL APPROACH TO QUALITY ENSURES LONG-TERM SUSTAINABILITY AND SCALABILITY.
- DEVELOPERS MUST BALANCE THESE FACTORS TO OPTIMIZE PERFORMANCE WITHOUT COMPROMISING USABILITY.

CLASSIFICATION OF QUALITY FACTORS

SOFTWARE QUALITY FACTORS ARE CATEGORIZED INTO THREE MAIN GROUPS: **PRODUCT OPERATION FACTORS, PRODUCT REVISION FACTORS, AND PRODUCT TRANSITION FACTORS.** EACH CATEGORY IMPACTS DIFFERENT ASPECTS OF SOFTWARE PERFORMANCE AND MAINTAINABILITY.

- **PRODUCT OPERATION FACTORS**:

  CORRECTNESS, RELIABILITY, EFFICIENCY, INTEGRITY, USABILITY.

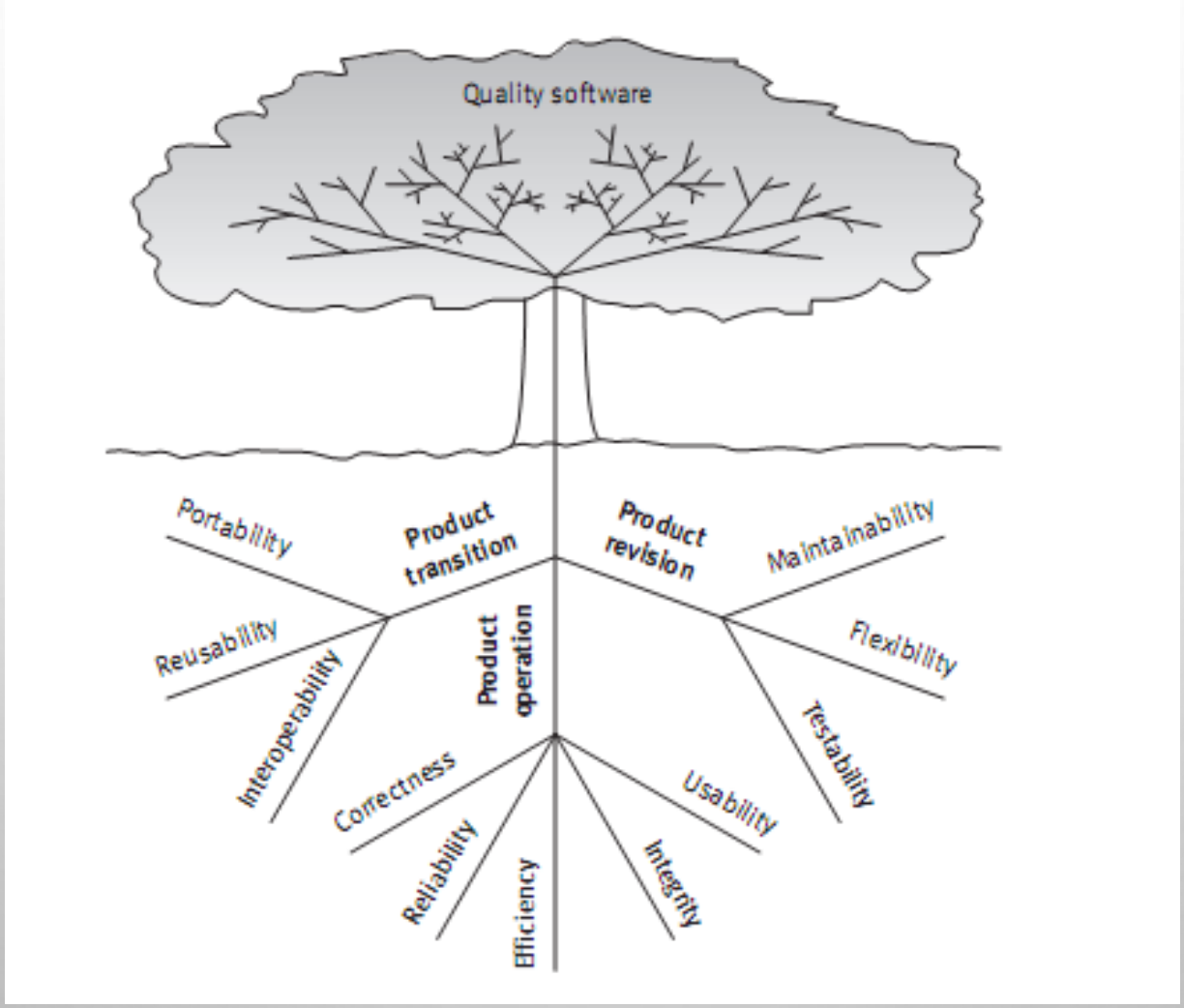
- **PRODUCT REVISION FACTORS**:

  MAINTAINABILITY, FLEXIBILITY, TESTABILITY.

- **PRODUCT TRANSITION FACTORS**:

  PORTABILITY, REUSABILITY, INTEROPERABILITY.

MCCALL'S FACTOR MODEL TREE



PRODUCT OPERATION FACTORS

- **CORRECTNESS**
  - REFERS TO THE DEGREE TO WHICH SOFTWARE MEETS ITS SPECIFIED REQUIREMENTS.
  - ENSURES THAT THE SOFTWARE PERFORMS THE INTENDED FUNCTIONS ACCURATELY.
  - EXAMPLE: A BANKING APPLICATION MUST CORRECTLY CALCULATE TRANSACTION FEES WITHOUT ERRORS.
  - TECHNIQUES TO ENSURE CORRECTNESS:
    - **FORMAL VERIFICATION, RIGOROUS REQUIREMENT ANALYSIS, AND AUTOMATED VALIDATION TESTING.**

PRODUCT OPERATION FACTORS

- CORRECTNESS REQUIREMENTS ARE DEFINED IN A LIST OF THE SOFTWARE SYSTEM'S REQUIRED OUTPUTS.
  - OUTPUT MISSION.
  - ACCURACY.
  - PRODUCT OPERATION COMPLETENESS.
  - UP-TO-DATE.
  - AVAILABILITY (RESPONSE TIME).
  - CODING AND DOCUMENTATION GUIDELINES.

PRODUCT OPERATION FACTORS

- **RELIABILITY** MEASURES THE SOFTWARE'S ABILITY TO FUNCTION WITHOUT FAILURE UNDER SPECIFIED CONDITIONS FOR A DEFINED PERIOD.
    - FACTORS INFLUENCING RELIABILITY: FAULT TOLERANCE, REDUNDANCY MECHANISMS, AND FAILURE RECOVERY.

RELIABILITY REQUIREMENTS ***DEAL WITH FAILURES*** TO PROVIDE SERVICE. THEY DETERMINE THE MAXIMUM ALLOWED SOFTWARE SYSTEM FAILURE RATE, AND CAN REFER TO THE ENTIRE SYSTEM OR TO ONE OR MORE OF ITS SEPARATE FUNCTIONS.

- SUB-FACTORS:
    - APPLICATION RELIABILITY.
    - HARDWARE FAILURE RECOVERY.
- STRATEGIES TO IMPROVE RELIABILITY:
    - **IMPLEMENTING AUTOMATED BACKUPS, REDUNDANCY, AND FAILOVER SYSTEMS.**

PRODUCT OPERATION FACTORS

EXAMPLES ON RELIABILITY

- THE FAILURE FREQUENCY OF A HEART-MONITORING UNIT THAT WILL OPERATE IN A HOSPITAL'S INTENSIVE CARE WARD IS REQUIRED TO BE LESS THAN ONE IN 20 YEARS. ITS HEART ATTACK DETECTION FUNCTION IS REQUIRED TO HAVE A FAILURE RATE OF LESS THAN ONE PER MILLION CASES.

- ONE REQUIREMENT OF THE NEW SOFTWARE SYSTEM TO BE INSTALLED IN THE MAIN BRANCH OF INDEPENDENCE BANK, WHICH OPERATES 120 BRANCHES, IS THAT IT WILL NOT FAIL, ON AVERAGE, MORE THAN 10 MINUTES PER MONTH DURING THE BANK'S OFFICE HOURS. IN ADDITION, THE PROBABILITY THAT THE OFF-TIME (THE TIME NEEDED FOR REPAIR AND RECOVERY OF ALL THE BANK'S SERVICES) BE MORE THAN 30 MINUTES IS REQUIRED TO BE LESS THAN 0.5%.

PRODUCT OPERATION FACTORS

- **EFFICIENCY**
  - FOCUSES ON THE OPTIMAL UTILIZATION OF SYSTEM RESOURCES SUCH AS CPU, MEMORY, AND NETWORK BANDWIDTH.
  - ENSURES THAT PERFORMANCE DOES NOT DEGRADE WITH INCREASED USER LOAD.
  - SUB-FACTORS: EFFICIENCY OF PROCESSING, EFFICIENCY OF STORAGE, EFFICIENCY OF COMMUNICATION, EFFICIENCY OF POWER USAGE (FOR PORTABLE UNITS).
- EXAMPLE: A REAL-TIME VIDEO STREAMING PLATFORM SHOULD MAINTAIN SMOOTH PLAYBACK WITH MINIMAL BUFFERING.
- METHODS TO ENHANCE EFFICIENCY:
  - **CODE OPTIMIZATION, CACHING TECHNIQUES, AND PERFORMANCE MONITORING.**

PRODUCT OPERATION FACTORS

- **INTEGRITY** DEFINES THE ABILITY OF SOFTWARE TO PREVENT UNAUTHORIZED ACCESS OR DATA CORRUPTION, AND IT ENCOMPASSES ASPECTS OF CYBERSECURITY, SUCH AS ENCRYPTION AND AUTHENTICATION.
- SUB-FACTORS: ACCESS CONTROL AND ACCESS AUDIT.
- EXAMPLE: AN E-COMMERCE WEBSITE MUST PROTECT USERS' PAYMENT INFORMATION THROUGH SECURE TRANSACTIONS.
- SECURITY MEASURES: **IMPLEMENTING MULTI-FACTOR AUTHENTICATION, ROLE-BASED ACCESS CONTROL, AND ENCRYPTION PROTOCOLS**.

PRODUCT OPERATION FACTORS

- **USABILITY** MEASURES HOW EASILY USERS CAN INTERACT WITH THE SOFTWARE.
  - FACTORS INCLUDE INTUITIVE INTERFACE DESIGN, ACCESSIBILITY, AND EASE OF LEARNING.
- EXAMPLE: A MOBILE BANKING APP WITH A SIMPLE, USER-FRIENDLY INTERFACE ENHANCES USER ADOPTION RATES.
- APPROACHES TO IMPROVE USABILITY: **CONDUCTING USER RESEARCH, USABILITY TESTING, AND ITERATIVE DESIGN IMPROVEMENTS.**

PRODUCT REVISION FACTORS

THESE FACTORS DETERMINE HOW EASILY SOFTWARE CAN BE MODIFIED OR IMPROVED AFTER DEPLOYMENT.

- **MAINTAINABILITY** REFERS TO THE EASE WITH WHICH SOFTWARE CAN BE UPDATED, DEBUGGED, OR EXTENDED.
  - REQUIRES WELL-STRUCTURED CODE, PROPER DOCUMENTATION, AND MODULAR DESIGN.
- EXAMPLE: OPEN-SOURCE SOFTWARE LIKE LINUX BENEFITS FROM HIGH MAINTAINABILITY DUE TO MODULAR ARCHITECTURE.
- BEST PRACTICES FOR MAINTAINABILITY: **ADOPTING CODING STANDARDS, USING VERSION CONTROL, AND WRITING CLEAN, DOCUMENTED CODE.**

PRODUCT REVISION FACTORS

- **FLEXIBILITY** MEASURES HOW EASILY SOFTWARE CAN BE ADAPTED TO NEW REQUIREMENTS OR ENVIRONMENTS.
  - INVOLVES DESIGNING SOFTWARE WITH CONFIGURABLE PARAMETERS AND MINIMAL HARDCODED VALUES.
  - SUB-FACTORS: MODULARITY, GENERALITY, SIMPLICITY, SELF-DESCRIPTIVENESS.
- EXAMPLE: A CLOUD-BASED CRM SYSTEM SHOULD ALLOW BUSINESSES TO CUSTOMIZE WORKFLOWS BASED ON CHANGING NEEDS.
- TECHNIQUES FOR FLEXIBILITY: **USING APIS, DESIGNING SOFTWARE WITH MICROSERVICES ARCHITECTURE, AND IMPLEMENTING CONFIGURATION FILES.**

EXAMPLES ON FLEXIBILITY

TSS (TEACHER SUPPORT SOFTWARE) DEALS WITH THE DOCUMENTATION OF PUPIL ACHIEVEMENTS, THE CALCULATION OF FINAL GRADES, THE PRINTING OF TERM GRADE DOCUMENTS, AND THE AUTOMATIC PRINTING OF WARNING LETTERS TO PARENTS OF FAILING PUPILS. THE SOFTWARE SPECIFICATIONS INCLUDED THE FOLLOWING FLEXIBILITY REQUIREMENTS:

- THE SOFTWARE SHOULD BE SUITABLE FOR TEACHERS OF ALL SUBJECTS AND ALL SCHOOL LEVELS (ELEMENTARY, JUNIOR AND HIGH SCHOOLS).

- NON-PROFESSIONALS SHOULD BE ABLE TO CREATE NEW TYPES OF REPORTS ACCORDING TO THE SCHOOLTEACHER'S REQUIREMENTS AND/OR THE CITY'S EDUCATION DEPARTMENT DEMANDS.

PRODUCT REVISION FACTORS

- **TESTABILITY** DEFINES HOW EASILY SOFTWARE CAN BE TESTED TO IDENTIFY DEFECTS AND ENSURE FUNCTIONALITY.
  - REQUIRES WELL-DEFINED TEST CASES, AUTOMATED TESTING FRAMEWORKS, AND CLEAR REQUIREMENTS.
- SUB-FACTORS: USER TESTABILITY, FAILURE MAINTENANCE TESTABILITY, TRACEABILITY.
- EXAMPLE: UNIT TESTS IN AGILE DEVELOPMENT HELP MAINTAIN SOFTWARE QUALITY THROUGHOUT ITERATIONS.
- TOOLS FOR TESTABILITY: **SELENIUM, JUNIT, TESTNG, AND AUTOMATED CI/CD PIPELINES.**

PRODUCT TRANSITION FACTORS

THESE FACTORS DEFINE THE SOFTWARE'S ABILITY TO ADAPT TO DIFFERENT ENVIRONMENTS AND INTEGRATE WITH OTHER SYSTEMS.

- **PORTABILITY** MEASURES HOW EASILY SOFTWARE CAN BE TRANSFERRED FROM ONE ENVIRONMENT TO ANOTHER.
  - REQUIRES PLATFORM-INDEPENDENT CODE, COMPATIBILITY LAYERS, AND MINIMAL DEPENDENCIES.
  - SUB-FACTORS: SOFTWARE SYSTEM INDEPENDENCE, MODULARITY, SELF DESCRIPTIVE.
- EXAMPLE: A WEB APPLICATION SHOULD RUN SEAMLESSLY ON DIFFERENT BROWSERS AND OPERATING SYSTEMS.
- STRATEGIES FOR PORTABILITY: **WRITING CROSS-PLATFORM CODE USING FRAMEWORKS LIKE ELECTRON OR FLUTTER.**

PRODUCT TRANSITION FACTORS

- **REUSABILITY** DEFINES HOW MUCH OF THE SOFTWARE'S CODE OR COMPONENTS CAN BE REUSED IN DIFFERENT APPLICATIONS.
  - ENCOURAGES MODULAR PROGRAMMING, OBJECT-ORIENTED PRINCIPLES, AND USE OF FRAMEWORKS.
- SUB-FACTORS: MODULARITY, DOCUMENT ACCESSIBILITY, SOFTWARE SYSTEM INDEPENDENCE, APPLICATION INDEPENDENCE, SELF DESCRIPTIVE, GENERALITY, SIMPLICITY.
- EXAMPLE: SOFTWARE LIBRARIES LIKE TENSORFLOW ALLOW DEVELOPERS TO REUSE AI/ML FUNCTIONS ACROSS DIFFERENT PROJECTS.
- ENHANCING REUSABILITY: **USING DESIGN PATTERNS, MODULAR COMPONENTS, AND CODE REFACTORING.**

PRODUCT TRANSITION FACTORS

- **INTEROPERABILITY** ENSURES SOFTWARE CAN EXCHANGE DATA AND OPERATE WITH OTHER SYSTEMS.
  - REQUIRES STANDARD COMMUNICATION PROTOCOLS (E.G., REST APIS, XML, JSON).
- EXAMPLE: A HEALTHCARE MANAGEMENT SYSTEM SHOULD INTEGRATE WITH HOSPITAL DATABASES AND MEDICAL DEVICES.
- STANDARDS TO ENSURE INTEROPERABILITY: USING STANDARDIZED DATA FORMATS AND MIDDLEWARE SOLUTIONS.

IMPORTANCE OF SOFTWARE QUALITY FACTORS

- **ENSURING HIGH SOFTWARE QUALITY LEADS TO:**
  - **ENHANCED USER SATISFACTION**
  - **REDUCED MAINTENANCE COSTS**
  - **COMPLIANCE WITH INDUSTRY STANDARDS**
  - **SCALABILITY AND ADAPTABILITY**
- EXAMPLES OF HIGH-QUALITY SOFTWARE:
  - GOOGLE CHROME (EFFICIENCY, USABILITY)
  - MICROSOFT AZURE (RELIABILITY, SCALABILITY)
  - TESLA AUTOPILOT (INTEGRITY, CORRECTNESS)

FACTORS OF THE ALTERNATIVE MODELS

- **MANAGEABILITY:** MANAGEABILITY REQUIREMENTS REFER TO THE ADMINISTRATIVE TOOLS THAT SUPPORT SOFTWARE MODIFICATION DURING THE SOFTWARE DEVELOPMENT AND MAINTENANCE PERIODS, SUCH AS CONFIGURATION MANAGEMENT, SOFTWARE CHANGE PROCEDURES, AND THE LIKE.

WHO IS INTERESTED IN THE DEFINITION OF QUALITY REQUIREMENTS?

- SOME QUALITY FACTORS NOT INCLUDED IN THE TYPICAL CLIENT'S REQUIREMENTS DOCUMENT MAY, IN MANY CASES, INTEREST THE DEVELOPER. THE FOLLOWING LIST OF QUALITY FACTORS USUALLY INTEREST THE DEVELOPER WHEREAS THEY MAY RAISE VERY LITTLE INTEREST ON THE PART OF THE CLIENT:
  - **PORTABILITY.**
  - **REUSABILITY.**
  - **INTEGRATION.**

SOFTWARE COMPLIANCE WITH QUALITY FACTORS

- QUALITY FACTORS IS EXAMINED BY DESIGN REVIEWS, SOFTWARE INSPECTIONS, SOFTWARE TESTS, AND SO FORTH.
- COMPREHENSIVE DISCUSSIONS OF DESIGN REVIEWS, SOFTWARE TESTING, SOFTWARE QUALITY METRICS AND OTHER TOOLS FOR VERIFYING AND VALIDATING THE QUALITY OF SOFTWARE ARE PROVIDED IN THE BALANCE OF THIS BOOK.

CONCLUSION

- SOFTWARE QUALITY FACTORS ARE ESSENTIAL FOR BUILDING ROBUST AND SCALABLE APPLICATIONS. PRIORITIZING THEM ENSURES LONG-TERM SUCCESS.