Course 4

Software Quality Assurance
&
Software Quality Models

Software quality assurance

- IEEE definition:

  1. A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements.

  2. A set of activities designed to evaluate the process by which the products are developed or manufactured. Contrast with quality control.

SQA – extended [Galin]

A systematic, planned set of actions necessary to provide adequate confidence that the software development process or the maintenance process of a software system product conforms to established functional technical requirements as well as with the managerial requirements of keeping the schedule and operating within the budgetary confines

	IEEE	Extended
Systematic, planned actions	✓	✓
Software Development Process	✓	✓
Software Maintenance	✗	✓
Functional Technical Requirements	✓	✓
Scheduling	✗	✓
Budget control	✗	✓

SQ Assurance vs. SQ Control

- Systematic activities throughout the development and maintenance – prevent, detect, correct errors
- Minimize costs of quality

- Set of activities – reject products that do not qualify
- Part of SQ assurance activity

Objectives of SQA – process oriented

1. Software will conform to functional technical requirements

3. Software will conform to managerial scheduling and budgetary requirements

5. Improvement and efficiency of software development and SQA – improve technical and managerial requirements + reduce costs

Objectives of SQA – product oriented

1. Software maintenance activities will conform to functional technical requirements

3. Software maintenance activities will conform to managerial scheduling and budgetary requirements

5. Improvement and efficiency of software maintenance and SQA – improve technical and managerial requirements + reduce costs

Software Quality Models

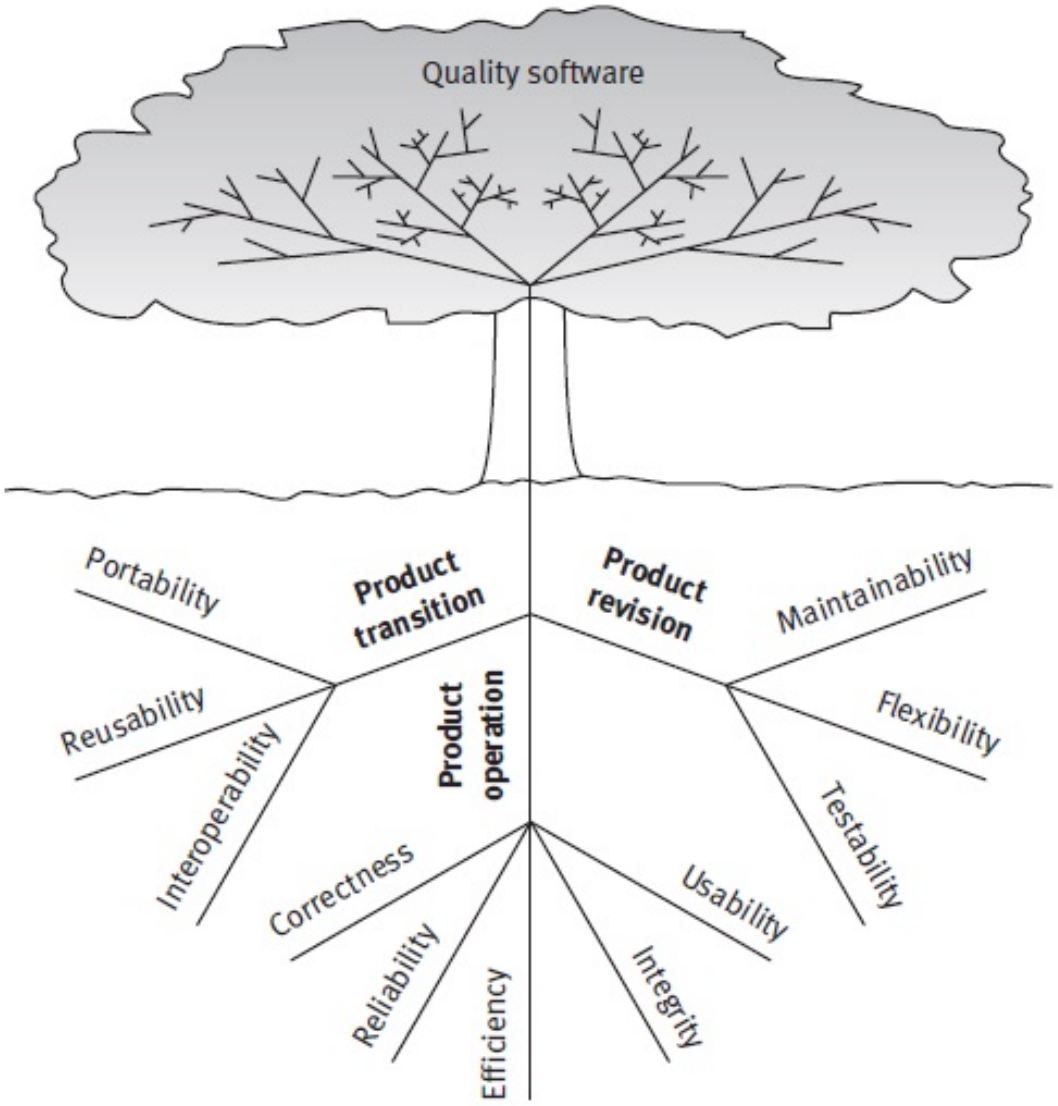
- McCall (1977) – classic – 11 factors
- Evans & Marciniak (1987) – 12 factors
- Deutsch & Willis (1988) – 15 factors
- ISO 9126 (2001) – 6 factors
- ISO25010 (2011) – 8 factors

Terminology

- Hierarchical structure: set of factors with subfactors
- Factor/ subfactor = characteristics / subcharacteristics
- In McCall model – same subfactor part of different factors
- Rest of the models: tree structure

McCall Model

- 11 factors – grouped in 3 categories:
  - Product operation factors - deals with requirements that directly affects software operation: Correctness, Reliability, Efficiency, Integrity, Usability
  - Product revision factors - deals with requirements affecting software maintenance activities: Maintainability, Flexibility, Testability
  - Product transition factors - deals with requirements affecting adaptation and integration: Portability, Reusability, Interoperability


Quality software
Portability
Reusability
Interoperability
Product transition
Product operation
Correctness
Reliability
Efficiency
Integrity
Usability
Testability
Flexibility
Maintainability
Product revision

Other models

Evans & Marciniak

- Exclude testability
- Add verifiability
- Add expandability

Deutsch & Willis

- Exclude testability
- Add verifiability
- Add expandability
- Add safety
- Add manageability
- Add survivability

➢Expandability + survivability – resemble flexibility + reliability
➢Testability – part of maintainability

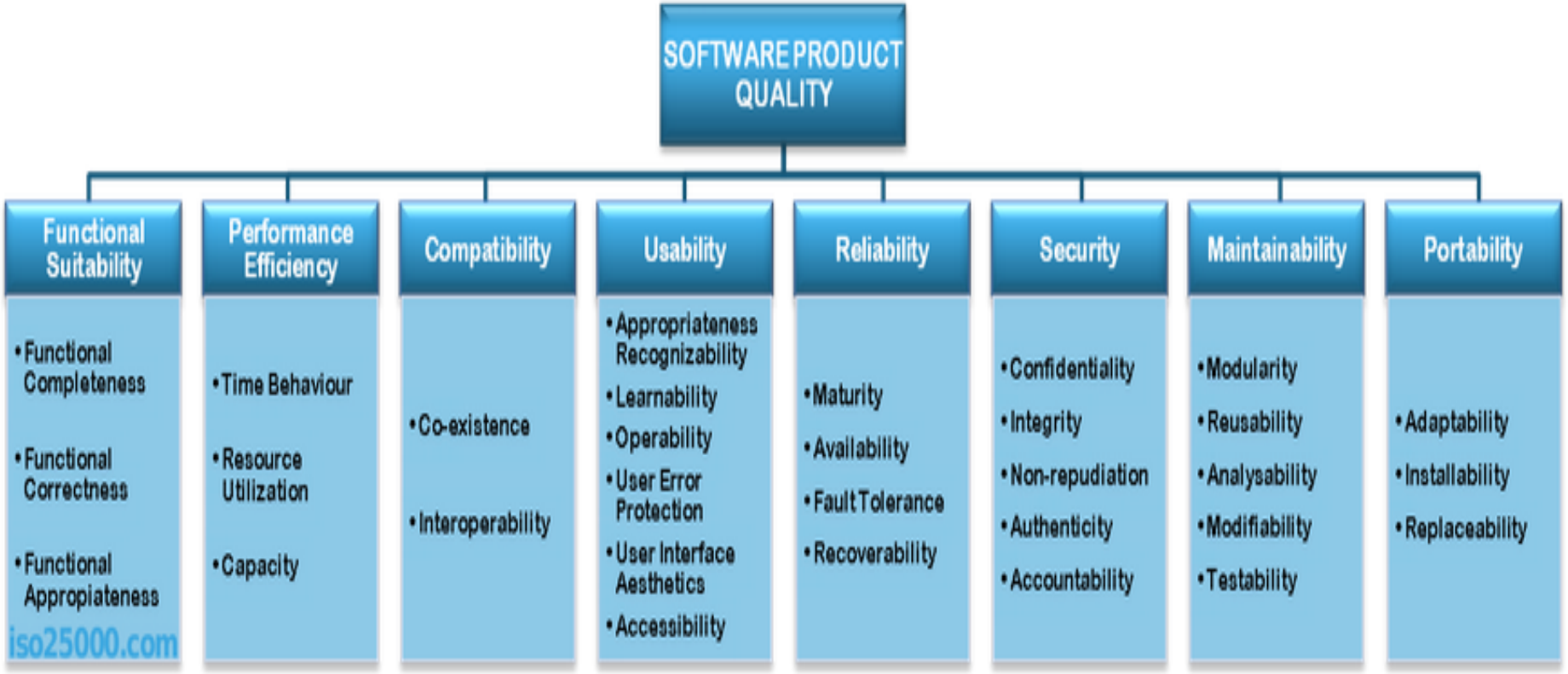
FURPS Model

- Developed by Hewlett Packard
- FURPS:
  - Functionality - is assessed by evaluating the features and capabilities of the delivered program and the overall security of the system.
  - Usability - is assessed by considering human factors, overall aesthetics, look and feel and easy of learning.
  - Reliability - is assessed by measuring the frequency of failure, accuracy of output, the mean-time-to-failure(MTTF), ability to recover from failure.
  - Performance - is assessed by processing speed, response time, resource utilization, throughput and efficiency.
  - Supportability - is assessed by the ability to extend the program (extensibility), adaptability, serviceability and maintainability.

ISO 9126 Quality Factors

- The ISO 9126 standard identifies six key quality attributes:
  - Functionality - degree to which software satisfies stated needs.
  - Reliability - the amount of time the software is up and running.
  - Usability - the degree to which a software is easy to use.
  - Efficiency - the degree to which software makes an optimum utilization of the resources.
  - Maintainability - the ease with which the software can be modified.
  - Portability - the ease with which a software can be migrated from one environment to the other.

ISO 25010
System and software quality model



McCall model

- See McCall.pdf in course directory
- Approach:
  - Determine a set of quality factors
  - Develop a set of criteria for each factor
  - Define metrics for each criterion
  - Validate metrics
  - Translate results into guidelines
- Start with ≈55 features - group

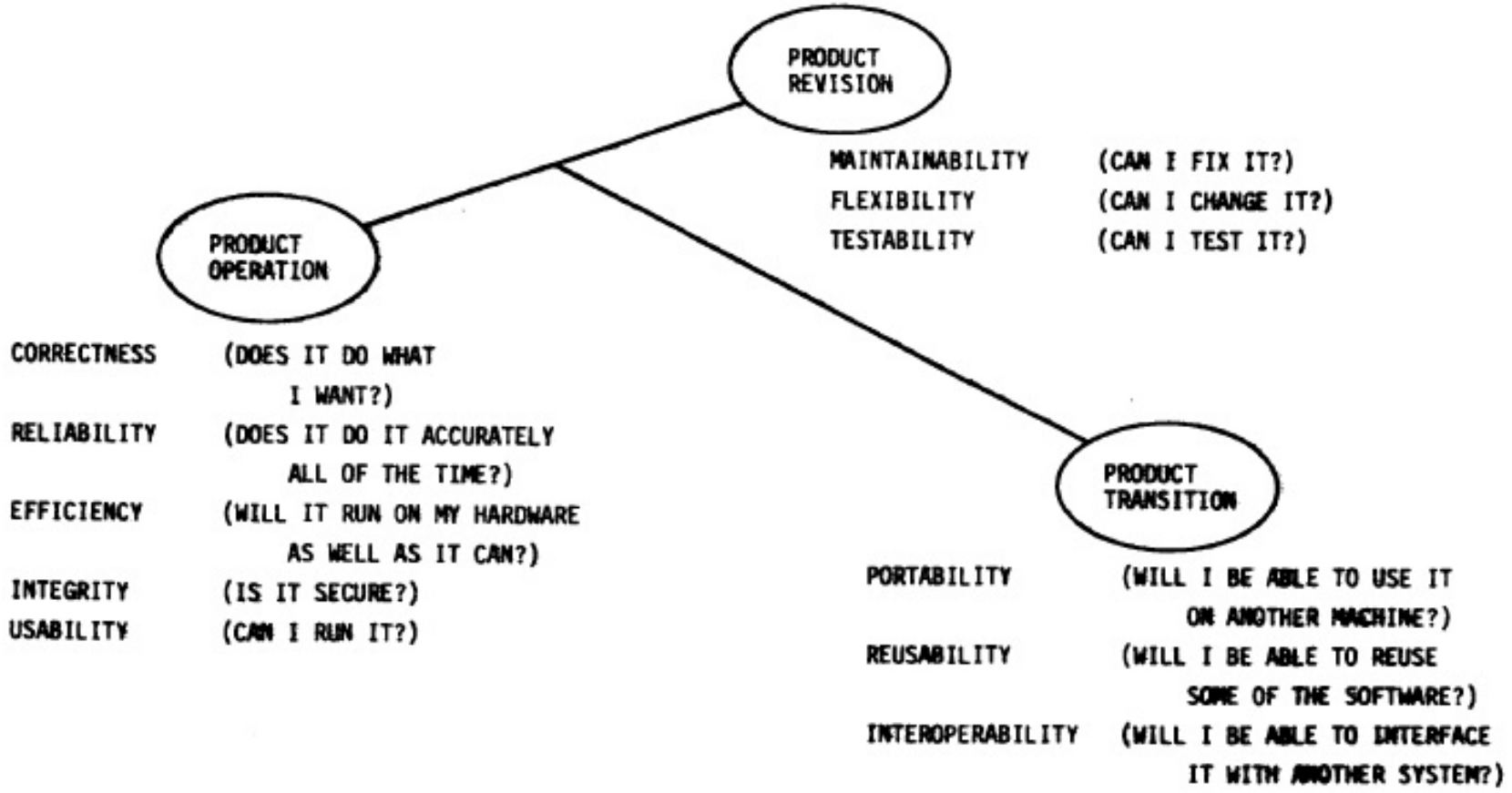


Figure 3.1-1 Allocation of Software Quality Factors to Product Activity 1352

Case study - Correctness

- Definition: Extent to which a program satisfies its specifications and fulfills the user's mission objectives.
- Life-cycle involvement:
  - Measured in development: analysis, design, implementation
  - Impact realized in: testing, operation, maintenance
- Criteria:
  - Traceability
  - Consistency
  - Completeness
- Criteria definition
- Metrics & criteria evaluation

- Traceability: Those attributes of the software that provide a thread from the requirements to the implementation with respect to the specific development and operational environment.

- Consistency: Those attributes of the software that provide uniform design and implementation techniques and notation.
  - Reliability + Maintainability

- Completeness: Those attributes of the software that provide full implementation of the functions required.

Metrics for criteria

- McCall.pdf:
  - pg 64 – Computation
  - pg 90 - Explanation

- Project assignment:
  - 1st step: choose your SQ model

PORTABILITY

Portability

- Definition: Effort required to transfer a program from one hardware configuration and/or software system environment to another.

- Impact:
  - Measured:
    - Design
    - Code
  - Realized: transition

McCall

Portability

- Modularity
  - struct. of indep. modules
  - in other 6 factors
- Self-descriptiveness
  - explanations
  - Also in maintainability, testability, portability & reusability
- Machine Independence
  - hardware indep.
  - also in reusability
- Software System Independence
  - indep. of OS, utilities, I/O
  - also in reusability

ISO25010

Portability

Adaptability

Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments

Instability

Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment

Replaceability

Degree to which a product can replace another specified software product for the same purpose in the same environment.

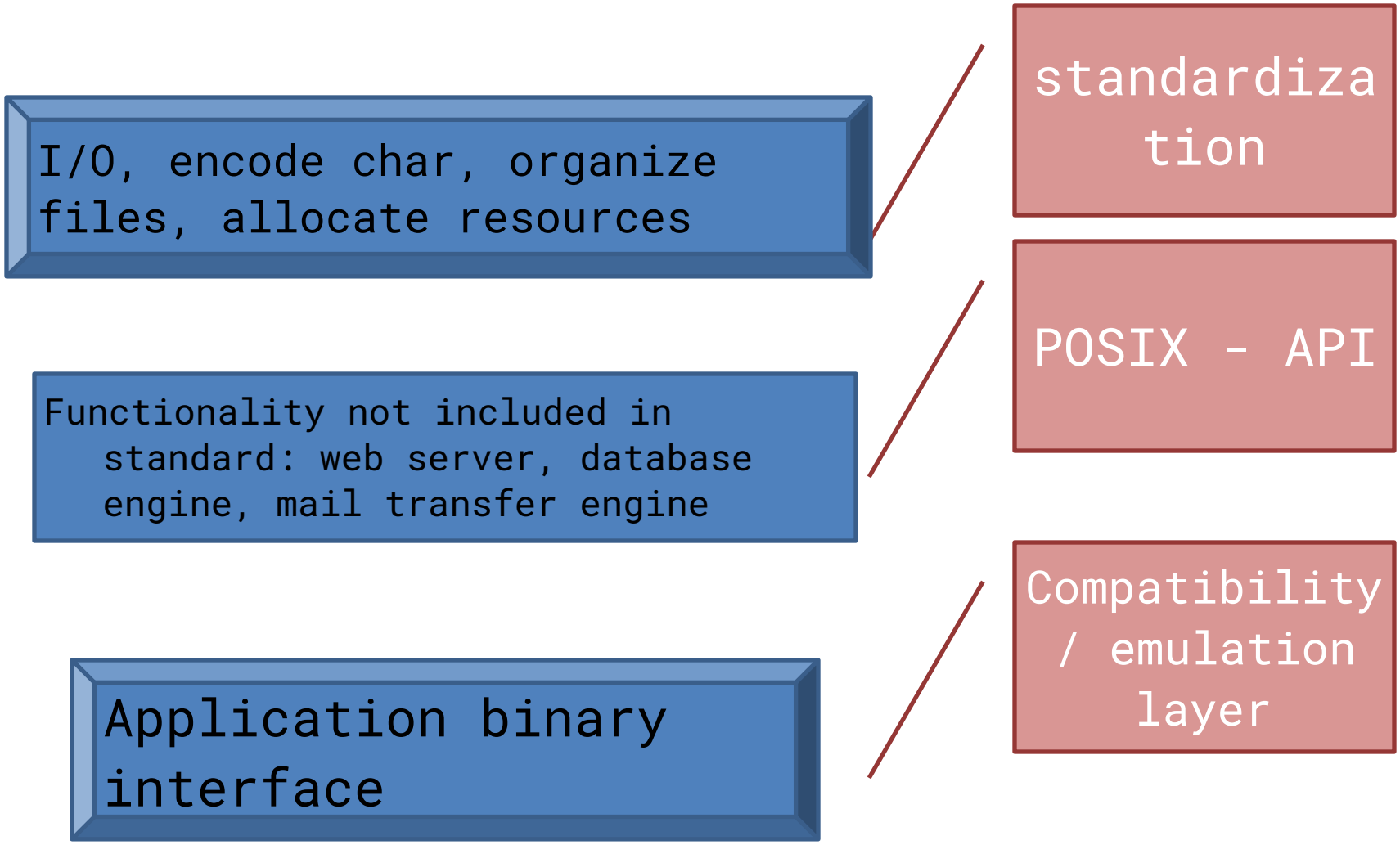
Why?

- Software life ≈15 years vs. hardware life ≈ 4-5 years
- Software implemented ≈ 3 hardware config.
- Porting less expensive than implementing
- Greater market

The 7 dimensions of portability

1. Operating systems
2. Processor architecture
3. Compiler & language features
4. GUI environment
5. Regions
6. Hardware devices
7. …

Issues in OS portability



Issues in Processor Architecture portability

- Data type properties – size + operations
- Data representation – alignment
- Machine-specific code

JAVA

Approaches in GUI portability

- Ignore: platform dependent app.
- Emulation layer: use libraries for transformation
- Portability layer: isolate GUI elem.
- Portable platform: ex. Java – own API for GUI
- HTML or AJAX-based layer

Issues in region portability

- Internationalization: generalization process of creating programs that can be easily ported across different regions
- Localization: particularization effort required to port a program to a given region

Portability Today

SOA

- Less technology dependent
- Internal platform independence through SOA

Portability today

Open Source

- Less coding, more composition
- Portability of parts

Portability metrics