1991 – 1992 ACADEMIC TRAINING PROGRAMME
Lectures series for Postgraduate students

SPEAKER : R. PRESSMANN / USA
TITLE : The present status of software engineering
TIME : 4 & 5 November from 10.30 to 12.00 hrs
PLACE : 4 November : Auditorium
5 November : Council Chamber

ABSTRACT

In this seminar, we will discuss the present status and future directions of software engineering and CASE. Key topics to be discussed include: new paradigms for software engineering; software metrics; process assessment; the current state of analysis and design methods; reusability and re-engineering; formal methods.

Among the questions to be answered are:

- How will software engineering change as the 1990s progress?
- What are the 'technology drivers'?
- What will analysis, design, coding, testing, quality assurance and software management look like in the year 2000?
- How will CASE tools evolve in the 1990s and will they be as 'integrated' as many people believe?
- How can you position your Organization to accommodate the coming changes?
The Future of Software Engineering & CASE: New Paradigms, Methods and Tools

Roger S. Pressman, PhD
R.S. Pressman & Associates, Inc.

OUR GOAL IN THE 1990s AND BEYOND
SOFTWARE ENGINEERING IN THE '90S

- paradigms will change more slowly than many expect, but they will change
- older methods will penetrate more shops, but at the same time, new methods, better notation and powerful supporting tools will be introduced
- tools will become integrated and will make the process easier to accomplish, but problems will remain

SOFTWARE ENGINEERING: WHAT IS THE TECHNOLOGY?
CURRENT PARADIGMS

- classic life cycle (the "waterfall" model)
  - a classical engineering approach
  - exemplified by DoD/Std 2167A
  - explicit start and end points
  - clearly defined milestones

- prototyping
  - creating a model/mock-up
  - emphasizes rapid development
    - 4GL, reuse, graphical tools
  - can partially replace the waterfall model
  - can completely replace the waterfall model

PARADIGMS FOR THE 1990s

- classic life cycle (the "waterfall" model)
- prototyping
- the spiral model
- the "reuse" model
- the "cleanroom" model
THE SPIRAL MODEL

THE REUSE MODEL
THE CLEANROOM MODEL

- uses one of the other paradigm structures, but ...
- quality is emphasized from the beginning
- formal methods for specification are used
- formal inspections are conducted
- proof of correctness is applied to all code that is developed
- statistically based independent testing is conducted

CURRENT METHODS

- hacking and handcuffing are widely used methods
- structured analysis has been boosted by CASE
- data design (E-R modeling) is making in-roads
- structured design is used for architectural design
- structured programming is the only widely used procedural design approach
- object-oriented design is gaining fast
- testing methods exist, but are not widely used
MANAGEMENT METHODS IN THE 90s

- customer and developer must negotiate requirements as a team
- measurement is essential and should be based on code independent criteria
- measurement leads to statistical evaluation of quality
- risk can be assessed formally using statistical techniques

TECHNICAL METHODS IN THE 90s

- object-oriented technologies are the hottest technical topic in software engineering
- formal methods promise higher quality software and the creation of programs at a different level of abstraction
- the maintenance burden continues to grow and reverse and re-engineering offer the first glimmer of automated support
CASE—CURRENT TOOLS

business systems planning
project management
support (DTP, network)
configuration management
analysis and design
simulation
prototyping
programming
integration & testing
re-engineering

DISILLUSIONMENT?

✓ "we're not seeing the productivity gains we expected ..."
✓ "my people say it's not "robust enough" whatever that means ..."
✓ "we don't need more documentation, we need more programs ..."
✓ "integration is bleep, the tools don't fit together and I think they create more work that they should ..."
BUT THE TECHNOLOGY SPREADS

- CASE awareness is higher than ever
- more importantly, management is beginning to loosen purse strings (out of desperation?)
- trend toward workstations continues, some disillusionment with mainframe development environments
- some (but not all) components of software engineering are being applied in 50-60 percent of all shops—a significant improvement over 10 years

WHAT CASE TOOLS NEED FOR THE 90s

- consistent notation for engineering models
  - analysis models, design models
- heavier design emphasis
  - real design rules, implementation guidance
- better implementation of technical metrics
  - complexity metrics, functional independence
- a multi-dimensional approach to engineering
  - relations that support top-down, bottom-up
- effective support for reusability
  - advanced browsers and "fuzzy" pattern matchers
CASE TOOLS FOR THE 1990s

- project management—emphasis on metrics collection
- support—greater emphasis on re-use through better SCM facilities, and ultimately, object-oriented environments
- analysis and design—tools with domain specific components that will enable semantic analysis of specifications and design
- programming—object-oriented languages and 4GLs

CASE TOOLS FOR THE 1990s

- testing—broader array of tools that enable automatic generation of test cases; better test management tools
- prototyping/simulation—models to code
- re-engineering—abstraction level and completeness will improve
- framework—will be dominated by large computer vendors and industry standards
CASE TOOLS

THE CHALLENGE:
PUTTING IT ALL TOGETHER
INTEGRATION OPTIONS

- simple data exchange
- common tool access
- control integration
- data integration
- full integration

DATA EXCHANGE

CASE data is pre- or postprocessed for translation to another tool (common interchange formats can be used)
COMMON TOOL ACCESS
An interface standard is used to give the tools the same look and feel. Data exchange is accomplished with pre- or postprocessing.

CONTROL INTEGRATION
Triggers enable one tool to notify other tools in the environment when significant events occur.
DATA INTEGRATION
Selected data is maintained in a shared repository, but tools each maintain their own private data.

FULL INTEGRATION

- User interface layer
  - Interface tool kit
  - Presentation protocol

- Tools management services

- Object management layer
  - Integration services
  - Configuration management services

- Shared repository layer
  - CASE database
  - Access control functions
DATA INTEGRATION: THE CASE REPOSITORY

REPOSITORY = PROJECT DATA + ENGINEERING DATA

Project data
- plans
- tasks
- schedule
- resource info.
- tool info.
- metrics

Engineering data
- Meta-data
  - programs
  - documents
  - data
  - versions
  - SCIs
- Technical data
  - design classes
  - rules
  - constraints
  - reusable code
  - data elements
ENVIRONMENT USERS
As an integrated environment evolves, the number and types of users will grow, leading to new demands on the repository and the environment.

THE MANAGEMENT LANDSCAPE: 2000AD
- evolutionary paradigms will be the norm
- metrics will be used to tune the process and improve the product
- AI-based risk analysis methods will aid with contingency planning
- re-engineering will be a strategic concern
- integrated CASE will enable managers to track projects more effectively
- BUT, the problems we know today will not disappear
THE TECHNICAL LANDSCAPE: 2000AD

- metrics will be used to assess technical quality
- re-engineering will follow a "fix when touched" rule
- object-oriented methods will flourish and account for 60-70% of all new work
  - reusable component libraries will be common
  - consequence: development time will be reduced
  - consequence: relatively little "from scratch" coding
- formal methods (with tools support) will make significant in-roads, particularly in safety critical systems
- domain specific AI-based tools will (finally) become real

SOFTWARE ENGINEERING IN 2000

paradigms

methods

tools

measurement
management
automation
technology
research
cultures
controversy

copyright © 1991 R.S. Pressman & Associates, Inc. 420 Eastslope Drive, Orange, Connecticut 06477 USA—phone (203) 795-5044