SOFTWARE ENGINEERING, PROSPECTS, PROBLEMS AND FUTURES

OR

what to do in the absence of an engineering discipline

(An Iconoclastic view)

by

Karl Reed, MSc, FACS,MIE(Aust)
Director, AAITP at La Trobe Univ.
Director, ACS Tech. Board (CS&SE),
Assoc. Prof. SE at La Trobe Univ.

(Seminar given at James Cook University,Townsville Australia July 15 1994. Almost identical to a seminar give at Lawrence Livermore National Laboratory, Livermore CA, January 1994. This note added 9/6/2011)

This Seminar will…

- ✏ *Argue that talk of an established discipline of Software Engineering is premature… this can be seen from a definition of SE.*
- ✏ *Identify a series of "missing links", items of methodology etc. which are required for software development to qualify as Engineering.*
- ✏ *Comment on impediments in industry, research and academia that are limiting progress towards this goal.*
- ✏ *Propose some steps which will address some of the problems,*
- ✏ *High light dangers for the established domains*

and do this in the time

Karl Reed, Director
Amdahl Itelligent
Tool Program LTU

Firstly, what is Engineering?

(This definition is a personal one…)

Practioners…

✏ *problem-solving mind-set.*

Using…

✏ *established techniques from theoretical disciplines…*

to design,build and maintain systems and artifacts…predictably & economically (iteratively)

Essential elements…differentiate between…

✏ *"problems" whose solution can be achieved using "prescribed" methods,*

and

✏ *situations where these methods do not appear to exist.*

✏ *processes must be repeatable*

Karl Reed, Director
Amdahl Itelligent
Tool Program LTU

MUCH ENGINEERING DESIGN KNOWLEDGE IS EMPIRICAL AND "RULE OF THUMB"

Why do we need Software Engineering?

- ✏ ***Too much software being written…(20B lines of new code per year?)***
- ✏ ***Costs of S/W inhibit wide-spread use of computers(?)***
- ✏ ***"Maintenance" costs too high (40-80% of developers budgets)***
- ✏ ***"Quality" of delivered code (errors, performance, reliability etc.) not adequate.***
- ✏ ***Software becoming pervasive, effecting all phases of our lives. (gives a new meaning to "safety critical" and environmentally sound).***

  US$5K BUYS A SYSTEM THAT, 30 YEARS AGO, WOULD HAVE SUPPORTED THE LARGEST COMPANIES, AND 50 PROGRAMMERS!

Why do we need Software Engineering?(cont'd)

✏ ***In general, unable to predict performance of s/w.***

(rely on improvements in h/w to solve problems)

✏ ***Unable to fix "appropriate" design scale!***

(Do we need a six-lane freeway or a footbridge to cross the creek? In S/W, either will do!)

- ✏ ***S/W IS AN ENABLING TECHNOLOGY…***
  increasing competativeness of existing industry…
  creating new industries…
- ✏ ***S/W IS A MAJOR INT'L COMMODITY…***
  WORLD MARKETS > US$80B
  Major national initiatives to promote product development in several countries…
  Japan…
  Europe…
  Singapore…
  Korea…
  Taiwan…

SOFTWARE PRODUCTION CAPABILITY A MAJOR ECONOMIC LEVER

WHAT IS SOFTWARE ENGINEERING?

THE IDENTIFICATION OF, DESIGN , CONSTRUCTION AND WHOLE OF LIFE CYCLE MAINTENANCE OF (SOCIALLY) USEFUL SYSTEMS …using…

- ✏ *Prescribed design and implementation techniques,*
- ✏ *Standardised components,*
- ✏ *Tools,*
- ✏ *Iteration of design to achieve **and determine** both functional and performance objectives,*

…to…

- ✏ *Predetermined resource and time constraints,*
- ✏ *Predetermined reliability, safety and environmental standards*

…in a…

- ✏ *Well managed fashion.**(Reed, after Fairley and with Zucconi)***

SOFTWARE ENGINEERING RESEARCH?

The development of new techniques to improve Software Engineering Practice…

We'll talk about the impediments to this later…

MAJOR PROBLEMS...

- ✏ CODIFICATION OF (EXISTING) KNOWLEDGE (ESSENTIALLY EXTENDED REUSE OF DESIGN AS WELL AS COMPONENTS)
- ✏ DEVELOPMENT OF VALIDATED DESIGN PROCEDURES
- ✏ DEVELOPMENT OF VALIDATED CONSTRUCTION PROCEDURES...leading to maintainable systems...

INCORPORATING

- ✏ IMPLEMENTATION RESOURCE AND TIME ESTIMATION
- ✏ RUN-TIME RESOURCE UTILIZATION AND PERFORMANCE PREDICTION
- ✏ REPEATABILITY AND TRACE-ABILITY OF DESIGN AND CONSTRUCTION

BASED UPON
STANDARDS AND ACCEPTED PRACTICE
FOLLOWED BY

- ✏ **TECHNOLOGY TRANSFER**

Some interesting problems…

1. ***Nature of S/W development, "theory" vs practice.***
2. ***Design "Prescriptions"***
3. ***Implications of "maintenance" and extendability for implementation techniques,***
4. ***S/W reuse and its impact on methodology,***
5. ***Component-Based design***
6. ***Philosophy of "design" and "architecture"***

Validated methods, tools and languages etc. are required!

Some interesting problems, and current status…(cont'd)

1. Nature of S/W development, "theory" vs practice.

Here we find an example of bad "scholarship", and poor research.

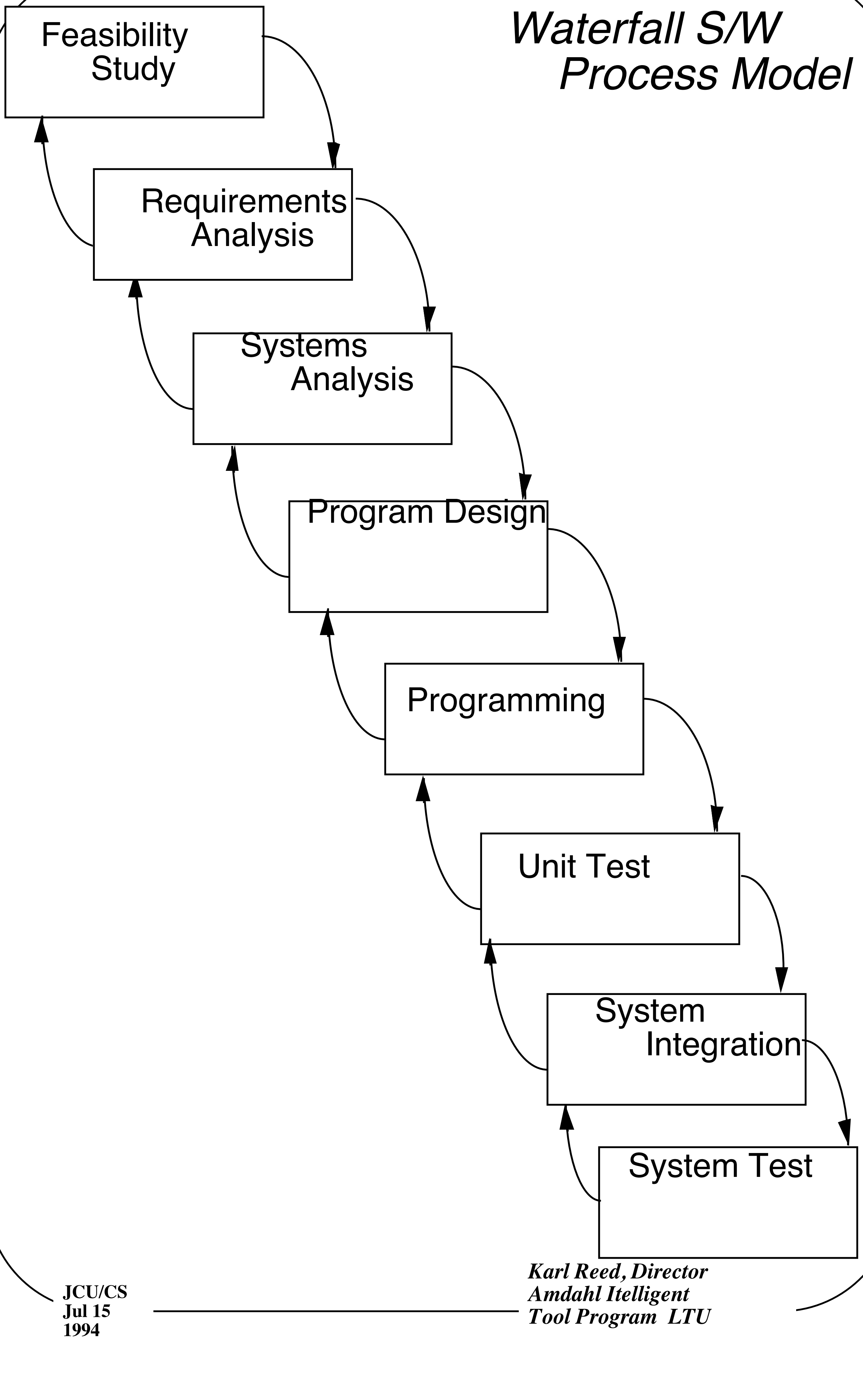
Consider the so-called "waterfall model", ascribed to Royce (1970).

There are two problems…

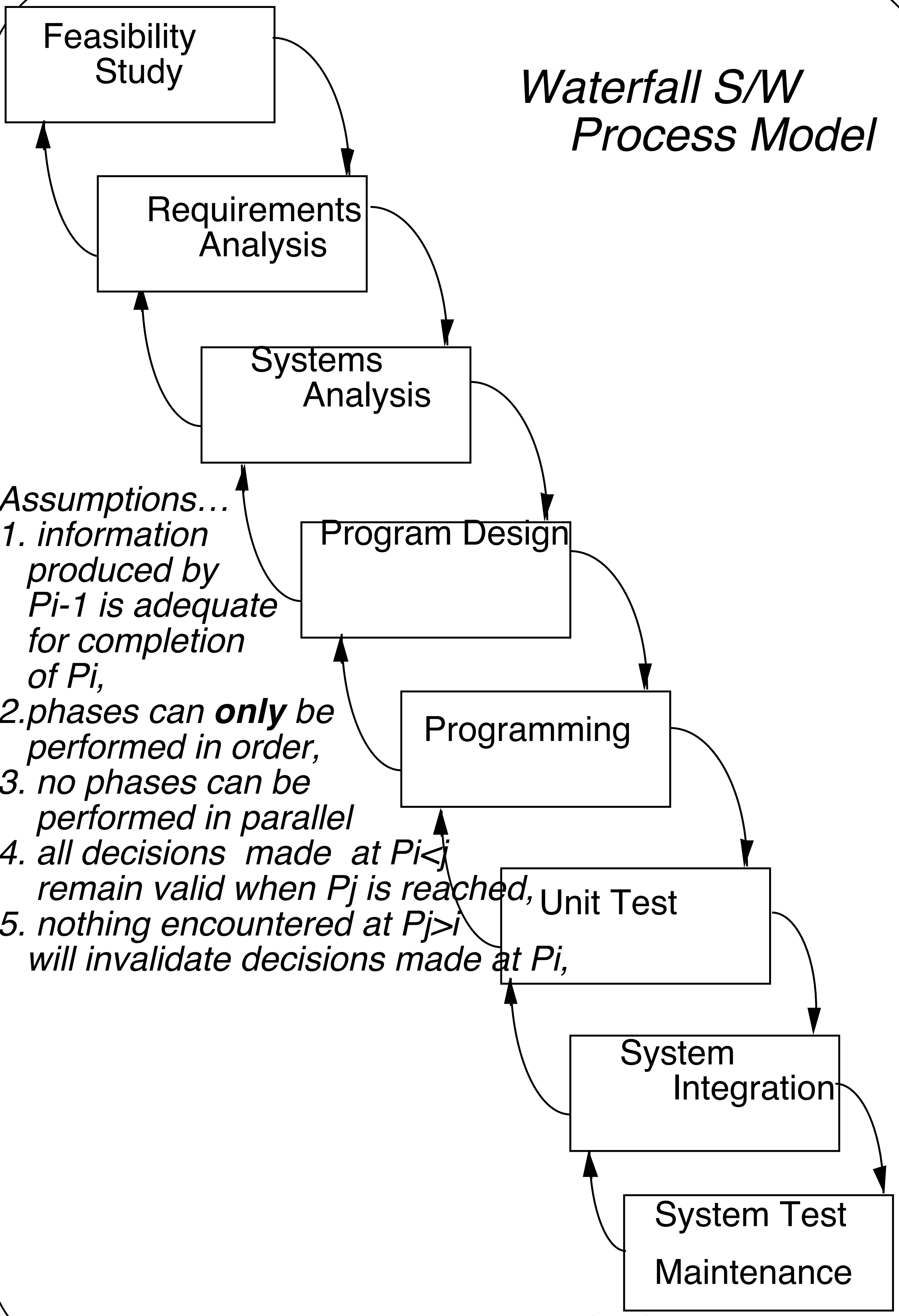
A. Is that what Royce actually said?

B. What are the implications of the model for s/w development?

Waterfall S/W Process Model



Waterfall S/W Process Model



Assumptions…
1. information produced by Pi-1 is adequate for completion of Pi,
*2.phases can **only** be performed in order,*
3. no phases can be performed in parallel
4. all decisions made at Pi<j remain valid when Pj is reached,
5. nothing encountered at Pj>i will invalidate decisions made at Pi,

Include slides showing Royce's model in detail, showing that prototyping and many other things are included

Waterfall Model… problems…

Assumptions…

1. information produced by Pi-1 is adequate for completion of Pi,

Implies types of diagrams, documents, text etc. reflect the intellectual material needed for completion of the next phase.

Some research, and anecdotal evidence contradicts this.

e.g. It is believed that productivity is high when ONE developer handles analysis thru unit-test.

Implies…

Need for improved diagraming and design-recording approaches.

Waterfall Model… problems…

2. phases can only be performed in order,

3. no phases can be performed in parallel

Implied by 1., but, assumes no knowledge ***acquisition*** *is required at $P_{j>i}$ which may be independent of P_i. If they need* ***not*** *be performed in order, then may be they can be performed in parallel.*

Experience shows that major problems can exist which are largely independent of the application specification, being heavily embedded in the implementation domains.(this has implications for a number of areas, e.g., s/w architecture, project design, etc.)

Waterfall Model… problems…

Assumptions…

4. all decisions made at Pi<j remain valid when Pj is reached,

5. nothing encountered at Pj>i will invalidate decisions made at Pi,

Implies…

No knowledge down stream of pi is needed by Pi, ***OR…***

If such knowledge exists, it will be apparent at Pi *(see problem 1.)*

Experience shows that this simply not true, and, that in the worst case, the original requirement for the system may be invalid by the time it is complete!

Some interesting problems, and current status…(cont'd)

Current developments in S/W process modelling…

Boehm's spiral model…
Osterweil's process programming(?)…
Prototyping…(not new, see Royce)…

Remaining issues…

Project Plan tailoring, i.e., how does one select a process model, and tailor it to a particular project?

Actual process models used in the real world?

THE FORMER IS ADDRESSED BY THIS PRESENTER,

THE LATER REQUIRES "DIRTY" WORK, I.E., THE STUDY OF ACTUAL S/W PRODUCTION… APART FROM VIC BASILI AND NASA/SEL, THERE AIN"T MUCH OF THAT ABOUT.

Some interesting problems…(cont'd)

2. Design "Prescriptions"

✏ *Existing methodologies tend not to give detailed procedures for moving from one "level" or "phase" to another.*

E.G. consider expanding a Data Flow Diagram to show more detail, or, a State Transition Diagram. The processes are "arbitrary" in the sense that there are many acceptable solutions, determined by developer's skill, prejudice, etc.

In contrast, other Engineering Disciplines (ED's) require calculations, use work-books,standard components & practice etc., ***RESTRICTING THE NUMBER OF POSSIBLE IMPLEMENTATIONS!***

E.G. a the design of a flange to join pipes of known diameter carrying gases at known pressure and flow.

which explains why no-one documents design…

Some interesting problems, …(cont'd)

2. Design "Prescriptions" Current developments…

§ Slow recognition of this fact, but, little explicit method. (Some in testing, O-O, for inheritance, but, otherwise little activity…)

§ Smith & Williams are working towards "engineering" approaches to performance engineering.

❁ ***Needs… Specific programs to identify missing links and develop prescriptive methods…AND VALIDATE THEM!***

An aside…

¶ Some research suggests that methodologies are not actually followed in practice…(Curtis, Siddiqi, Silverman)

If this is true, then…

- § Documentation occurs **after** the fact,
- § It cannot reflect the design process that actually occurred…
- § It will not aid maintainers (hence re-engineering)
- § **Implementers will not actually document design until they cannot complete the design without the documentation…**

In other words,
when the design process can

only be performed with calculations, explicit decisions, etc. then it will be documented!

Some interesting problems, …(cont'd)

3. Implications of "maintenance" and extendability for implementation techniques,

✏ *S/W is extensively modified…*

- *§ Specifications change,*
- *§ Customer requirements change,*
- *§ Delivery platforms change…*

We all know this!

To address this, we need…
DESIGN FOR MODIFIABLITY

Some interesting problems, …(cont'd)

3. DESIGN FOR MODIFIABLITY IN THIS CASE, THE TECHNOLOGY EXISTS, BUT NEEDS TO BE FORMALLY CONSOLIDATED INTO A PRESCRIPTIVE METHOD!

Technology available includes…

§ Table-driven,
§ Information hiding,
§ Exception handling,
§ "Come-From" statements…
§ State Transition Diagrams,
§ Interpretive techniques,
§ Application generators using compiler-writing techniques,

Some interesting problems, …(cont'd)

3. DESIGN FOR MODIFIABLITY

TO ACHIEVE THIS, WE CAN…

- ✏ *Invert the "Contraction of Generality" concept…*
- ✏ *Seek possible generalizations of function,*
- ✏ *Fabricate functions from common modules, using table driven approaches,*
- ✏ *Imbed logic in tables,*

ETC, ETC.

IN THIS CASE, WE HAVE A PROBLEM AND THE "CRUDE" TECHNOLOGY TO SOLVE IT…

SO, WHY HASN'T THIS BEEN DONE??

AT THIS POINT, ONE MIGHT ASK…

A. IF THE PROBLEMS AND BASE-TECHNOLOGY ARE KNOWN, WHY ISN'T THERE A METHOD?

B. IS THIS A FEATURE OF "SOFTWARE ENGINEERING"?

Yes, it seems that in many cases, this is true…

A number of researchers, (cf Harel), suggest that…

Many of our problems may have known solutions, if only we looked for them!

We are arguing that atleast we have have known problems… and can predict the nature of the solutions…

Lets return to our original theme…

Some interesting problems…

4. S/W reuse and its impact on methodology,

- ✏ ***WIDELY BELIEVED TO BE THE ONLY REAL HOPE…***
- ✏ ***SUBJECT OF CONSIDERABLE RESEARCH…***

PROBLEMS…

- § *Current methodologies do not generate reuseable components.* ***This requires iterative design and "late commitment" to component choices…***
- § *Current research tends to focus on sophisticated technology for reuse library management.* ***Experience with other "reuse" based disciplines suggests that simple, widely known libraries of components and designs***

Some interesting problems…

4. S/W reuse and its impact on methodology,

PROBLEMS…

- § *Need for frame-works for reuse, "plug-in" modules for s/w racks and frames…*
  Concepts exist in S/W bus and MLI's
- § *Organisational frame-works needed…*
  See Basili's reuse "factory" (I prefer the term "studio"
- § *reuse MUST extend to complete and partial designs…*

*Karl Reed, Director
Amdahl Itelligent
Tool Program LTU*

Some interesting problems…

4. S/W reuse and its impact on methodology,

PROBLEMS…

- *Current methodologies do not generate reuse-able components.* ***This requires iterative design and "late commitment" to component choices…***
- *Current research tends to focus on sophisticated technology for reuse library management.* ***Experience with other "reuse" based disciplines suggests that simple, widely known libraries of components and designs UNSUPPORTED BY COMPLEX TECHNOLOGY work very, very well***

Some interesting problems…

4. S/W reuse and its impact on methodology,

PROBLEMS…

- § *Need for frame-works for reuse, "plug-in" modules for s/w racks and frames…*
  Concepts exist in S/W bus and MLI's
- § *Organisational frame-works needed…*
  See Basili's Experience "factory" (I prefer the term "studio")
- § *reuse MUST extend to complete and partial designs…*
  There should be standardised designs, re-usable in systems development. Evidence is that these do in fact exist.

The ultimate goal here is "graphical" system composition...

Graphical icons representing components are "clipped" in to architecture/program diagrams to produce a system.

This is not really new, we do it with PC's and Mac's all the time???

Work is in progress in this area, e.g. program visualisation.

Some interesting problems…
4. S/W reuse and its impact on methodology,

There is significant work in this area, but, it still needs consolidation…

Some interesting problems…

5. Component-Based design

This is the other side of reuse, but addresses a different issue…

Design procedures assuming the prior existence of standardised components…

Difficulties are considered to exist here…

- ✏ *Absence of physical laws forcing component nature…software is infinitely malleable…*
- ✏ *Absence of encapsulation … s/w is (almost) infinitely accessible…*
- ✏ *Absence of speed/size benefits… s/w libraries (this need not be true…)*
- ✏ ***ABSENCE OF A COMPONENT-BASED DESIGN EDUCATION***

(warrants some explanation)

Some interesting problems…

5. Component-Based design

Addressing these issues…

S/W components …

- ✏ ***can and are being developed… commercial component libraries are now being marketed… C++***
- ✏ ***can be treated as immutable***
- ✏ ***Speed and size gains are possible***
- ✏ ***It is possible to train people to think in terms of component-based design… this is done for electronic engineers***
- ✏ *(component libraries were common in the late 1960's.Where did we go wrong?)*

MANY OF THE ISSUES MENTIONED SO-FAR ARE RELATED TO THE GENERAL ISSUES OF "REUSE" AND TO DESIGN…

Some interesting problems…

6. Philosophy of "design" and "architecture"

There is a deeper issue here…

What constitutes design in S/W?

Current "design" paradigms are primarily "working-result-oriented", i.e., so long as it works, the design is considered successful.

Constraint-based design (performance, reliability, quality, safety etc.) leads to uncharted territory.

How does the concept of architecture "fit" into s/w design?

(What is this anyway?)

Cox proposes a "product-centric design" paradigm, based on reuse.

Basili advocates an experience factory, re-using design at various levels… and goal-based adaption of methodology (cf GQM)

Reed advocates prescriptive design, "reasoning explicit"

6. Philosophy of "design" and "architecture"

Shaw and others draw attention to the concept of "architecture"…
… i.e. the existence of standard arrangements of sub-systems of known functions which can solve particular, generic problems.
Reed and others extend these concepts by proposing the existence of "rules" for connecting these components (implied by Shaw) which depend on their function and the system…
Perhaps architectural design can be delayed? (See comments on S/W process).
Can architecture be retrofitted?
What is it anyway?

Some interesting problems…

6. Philosophy of "design" and "architecture"

"Architecture" seems to involve…

- ✏ ***Standard arrangements of components…(Shaw identifies 5)***
- ✏ ***Various levels of reuse of design (cf "ordinary" architecture)***
  - § *Component design is achieved, hence is not performed, and has no effect on the final system.*
  - § *"" "" is known to be achievable, hence incompleteness is irrelevant, but may impact final system.*
- ✏ ***Concentration on aesthetic and functional requirements NOT performance, primarily***
- ✏ ***Rules for arranging components when standard architectures do not apply.***

Some interesting problems…

6. Philosophy of "design" and "architecture"

✏ ***Various levels of reuse of design (cf "ordinary" architecture)***

Component design…

§ is "completed", hence is not performed, and has no effect on the final system.

§ is known to be achievable, hence incompleteness is irrelevant, but may impact final system.

§ is known to be achievable, but may need to be completed to ensure final system is "correct".

§ ***is not known to be achievable …cf Sydney opera house.***

This can be understood easily in terms of standard building architecture.

Some additional major technical problems to be solved…(we need engineering methodologies, NOT simply new theory!)

- ✏ *Real knowledge of S/W production,*
- ✏ *"Powerful" formal methods, cf Laplace xforms v DE's(cf Parnas in TSE -9/93)*
- ✏ *Improved Diagraming Techniques,*
  - § ***Diagrams should be executable, cf. circuit diagrams etc.***
  - § ***Diagrams should reflect nature of (most) problems… async. communicating. parallel processes.***
- ✏ *"Efficient" test techniques*
- ✏ *"Testability" driven implementation techniques*
  - § ***Reduction in logical complexity of programs, perhaps through separation of control and data***

Impact of failure to study process and design procedures…
What have we got with CASE?

A pessimistic view…

- ✏ ***Automated methodologies which may not work… preventing work arounds.***

What might come from detailed studies of existing practice…

- ✏ ***Knowledge of existing design procedures,***
- ✏ ***Precise data on the useful knowledge used in the transition from one phase to to the next,*** *hence, some knowledge of the design documents which are useful,*
- ✏ ***Discovery of new process models, and rules for choosing them,***

AND, perhaps, the design of new tools, based on genuine CAD, implementing prescriptive procedures!

Given the problems, our ability to define the goals and the vast SE research effort…

Why aren't we making progress?

Briefly, the conservatism of the SE research community, and absence of under graduate SE degrees!

MAJOR PROBLEMS...(con'd)

- ✏ ***absence of undergraduate s.e. education, and relevant "mind-set".***
- ✏ ***academic appointment and promotion rules which mitigate against experience for teachers***
- ✏ ***failure to undertake "real" research issues***

The way forward…

THE SOLUTIONS TO THESE PROBLEMS REQUIRE....

✏ **STUDY OF EXISTING PRACTICE AND ARTIFACTS**

✏ **STUDY OF PAST PRACTICE AND ARTIFACTS**

✏ **DEVELOPMENT OF EXPERIMENTALLY VALIDATED ..**

- **§ PROGRAMMING STYLES AND LANGUAGES**
- **§ DESIGN AND CONSTRUCTION PROCEDURES**

these include diagraming systems, documentation techniques, tools, libraries, etc.

(FOLLOWING CCITT AND U.S. MILITARY AIRCRAFT PROCUREMENT MODELS)

✏ **APPROPRIATE EDUCATION**

MAJOR PROBLEMS...(CONT)

THE SOLUTIONS TO THESE PROBLEMS REQUIRE....

- ✏ **STUDY OF EXISTING PRACTICE**
- ✏ **STUDY OF PAST PRACTICE**

A NEW DISCIPLINE.....

SOFTWARE ARCHEOLOGY??

"WE ARE DISCARDING VALUABLE KNOWLEDGE AT AN ALARMING RATE, IGNORING THAT WHICH MAY BE USED BY FUTURE GENERATIONS, SIGNIFICANTLY RETARDING OUR RATE OF PROGRESS"

"What do the rubbish bins of the software shops of the world reveal?"

DIFFICULTIES FOR RESEARCHERS ....

✏ **NEED FOR ACCESS TO LARGE-SCALE S/W ARTIFACTS…**

- **EXISTING SYSTEMS**
- **TOOLS** ***(very expensive...)***

✏ **LEVEL OF EXPERIENCE OF RESEARCHERS..**

- ***"DEEP" PROBLEMS REQUIRING EXTENSIVE EXPERIENCE AND KNOWLEDGE AS WELL AS ABILITY***
- ***CONSOLIDATION OF KNOWLEDGE , A TASK FOR EXPERIENCED PERSONNEL***
- ***NATURE OF TRAINING OF NEW GRADUATES***

✏ **LEVEL OF EXPERIENCE OF SUPERVISORS**

- ***IN-DEPTH S/W CONSTRUCTION EXPERIENCE..NOT CONTINUOUS UNIV. BACKGROUND (we'll explore this further)***

DIFFICULTIES FOR RESEARCHERS (CONT)

✏ **PHD EXPECTATIONS**

- ***HIGH DEGREE OF ORIGINALITY, MUST NEVER HAVE BEEN DONE BEFORE! OFTEN NOVEL ONLY, EVEN GIMMICKY***
  - ***as a result, studies of existing systems and practice are not often pursued***
  - ***likewise codification of practice***

✏ **SCALE OF PROJECTS NEEDED IN MANY AREAS**

- ***MOVEMENT OF RESEARCH AREAS INTO INDUSTRIAL DOMAIN MEANS ACADEMIA MAY NOT BE ABLE TO COMPETE, (e.g. CASE, Hypertext)***

- ***CONTROLLED EXPERIMENTS ON LARGE SCALE PRODUCTION NEEDED TO VALIDATE CONCEPTS (e.g. design procedures, construction techniques, managerial, estimating, metrics, etc.)***

DIFFICULTIES FOR RESEARCHERS(CONT)

✏ SCALE OF PROJECTS NEEDED IN MANY AREAS (cont)

- *VALIDATION OF SYSTEM STRUCTURE PROPOSALS (distributed, network, o.s., etc)*

COMPUTER SCIENCE, THE DISCIPLINE OF UN-REPRODUCED RESULTS?

✏ MULTI-DISCIPLINARY STUDIES NEEDED

- *METRICS, ACTUAL SYSTEM DEVELOPMENT PRACTICE.*

SUMMARY...MAJOR PROBLEMS CANNOT BE SOLVED BY ACADEMIA, IN PRESENT STATE, AND WITH CURRENT AGENDAS

SOLUTION....ALTER AGENDAS...
.. FUND INDUSTRIAL STRENGTH RESEARCH, INSIDE ACADEMIA AND OUT...
.. NEW DOCTORAL DEGREE D.S.E. (?)
.. UND. GRADUATE S.E. DEGREES

DIFFICULTIES FOR RESEARCHERS (CONT)

EDUCATIONAL IMPLICATIONS

GIVEN THAT ACADEMIC RESEARCHERS ARE ALSO EDUCATORS, A MASSIVE CHANGE IS NEEDED IN THEIR APPOINTMENT AND PROMOTION CRITERIA IF THEY ARE TO BE ABLE TO CONTRIBUTE TO BOTH THE TRAINING OF SOFTWARE ENGINEERS AND TO RESEARCH ON THE CRITICAL PROBLEMS!!
ACADEMICS WITH EXPERIENCE WHO AID THE PROCESS OF RESOLVING THE MAJOR ISSUES MUST BE ENCOURAGED! (Gibbs, Reed, Shaw, Berry, and many others)

Conclusion
WE HAVE IDENTIFIED…

- ✏ Specific missing links which disqualify current SE as an Engineering Discipline identiied,
- ✏ General properties of Engineering Discipline Identified as applied to SE,
- ✏ Factors inhibiting progress noted,
- ✏ Absence of relevant research, except for NASA/SEL and a few others, noted…

HOW DO WE PROCEED?

the way forward…

- ✏ ***Establish s/w development process monitoring and improvement programs, with developers and academia,***
- ✏ ***Set research programs aimed at specific problems from those issues listed,***
  (recognize applied nature of the research, and the specific goals being set.)
- ✏ ***Create opportunities for experienced (20yr) vets to undertake research,***
- ✏ ***DEVELOP UNDERGRADUATE SE PROGRAMS, DETERMINED ENTIRELY BY AN ENGINEERING MIND-SET!***

THANK

YOU!

Some major Technical problems to be solved…(we need engineering methodologies, NOT simply new theory!)

- ✏ *Performance-constrained design techniques, (also, safety and reliability)*
- ✏ *Prescriptive design procedures, in which most steps are described precisely,*
- ✏ *Real knowledge of S/W production,*
- ✏ *"Powerful" formal methods, cf La Place xforms v DE's*
- ✏ *Improved Diagramming Techniques,*
- ✏ *System architecture and reuse philosophy and tools (i.e., libraries)*
- ✏ *"Efficient" test techniques*
- ✏ *"Testability" driven implemantation techniques*