Plan for the next few weeks

- **Mon 1 Oct**
  > Research paradigms and pro forma abstracts. Re-read Newman 94 and Shaw01, look at paper-writing advice on resources page
- **Wed 3 Oct, Mon 8 Oct, Wed 10 Oct**
  > Four research papers per class, 15-minute presentations
  > Choose a significant paper in your chosen topic area that illustrates both good results and good research strategy
  > Classic papers are prime candidates
- **Mon 15 Oct**
  > Plan projects, revisit framework for analysis. Re-read Redwine/Riddle
- **Wed 17 Oct for 11 classes**
  > 1 hour presentation/discussion of topic area
  > 1/2 hour other topic: classic paper, unify pro forma abstracts

Presentation Slots 10/3 to 10/10

- **Wed 3 Oct**
  > Shaw: Software Cost Estimation [Boehm97, supplemental Boehm84]
  > Tim Halloran: Configuration Management [Rochkind75, Tichy85]
  > Dean Sutherland: Slicing in Maintenance [Gallagher&Lyle91]
  > Annie Luo: Testing Tools [Hamlet&Voas <need citation>, supplemental Gelperin/Hetzel <need citation>]
- **Mon 8 Oct**
  > Elissa Newman: Program Slicing [Weiser81]
  > Owen Cheng:
  > George Fairbanks: Statemate [Harel87]
  > Bridgette Spitznagel:
- **Wed 10 Oct**
  > Vahe Poladian:
  > Beth Latronico:
  > Charles Shelton: Programming in the Large [DeRemer/Kron87]
  > Paul Li:
Executive Summary

- Technical ideas in software engineering evolve in predictable ways
- This evolution pattern sets expectations for software architecture research
- Lessons for developers: what to expect of software architecture research, now and in the future
- Lessons for researchers: useful research paradigms and validation techniques

Focus on Research

- Recall ...
  
  SE research is a prime source of the systematic knowledge
  SE research problems should reflect the constraints

- Turn now to research strategies
Research Strategy

Examine the kinds of research questions software engineers ask and the ways they study those questions

• Ideas mature over time
  They grow from qualitative and empirical understanding to precise and quantitative models

• Research questions are of different kinds
  Kinds of interesting questions change as ideas mature

• Research strategies also vary
  They should be selected to match the research questions

Research Objectives

Real World
Practical problem

Real World
Solution to practical problem

• Key objectives
  > Quality -- utility as well as functional correctness
  > Cost -- both of development and of use
  > Timeliness -- good-enough result, when it’s needed

• Address problems that affect practical software

Types of Research Questions

• Feasibility
  Does X exist, and what is it?
  Is it possible to do X at all?

• Characterization
  What are the characteristics of X?
  What, exactly, do we mean by X?
  What are the varieties of X, and how are they related?

• Method/Means
  How can we do X?
  What is a better way to do X?
  How can we automate doing X?

• Generalization
  Is X always true of Y?
  Given X, what will Y be?

• Discrimination
  How do I decide whether X or Y?

Parnas, “Modular Structure” - Questions

• ES?: Examination of real-life complex software projects has shown that information hiding principles have not been utilized by industrial software development teams. Authors have analyzed the possible reasons for lack of such adoption, and conclude that although applicable to small problems, the modularization of a software project based on information hiding does not work well with large complex software, where the number of modules is in the hundreds.

• ES? ET?: The effectiveness of information hiding in supporting the design of maintainable software systems has been argued (“demonstration” being declared the reason for the project the paper discusses)

• XH: The study reported here of modularization of a complex software system supported by information hiding...

• ET(presented as EM): The effectiveness of information hiding in supporting the design of software systems has been demonstrated.

• XH: Studies reported here of designing software systems supported by the principle of information hiding...

• ESNA: The technique of information hiding has been successfully applied to multiple small and straightforward academic software design problems, providing the major benefit of reduced cost by "allowing modules to be designed and revised independently." It is hoped that information hiding will provide similar benefits to problems of real-world complexity.

• ES: Studies of the use of information hiding through modularization have shown deficiencies in modularity in large-scale systems (of over 25 modules) because of information overload and unforeseen effects in the event of changes.
Research Strategy

Real World
Practical problem

Research Setting
Idealized problem

Real World
Solution to practical problem

Research product
(technique, method, model, system, …)

Types of Research Results

- Qualitative & descriptive models
  Report interesting observations
  Generalize from (real-life) examples

- Techniques
  Structure a problem area; ask good questions
  Invent new ways to do some tasks, including implementation techniques
  Develop ways to select from alternatives

- System
  Embody result in a system, using the system both for insight and as carrier of results
  Develop empirical predictive models from observed data
  Develop structural models that permit formal analysis

Maturity: Kinds of Research Results

Brooks proposed recognizing three kinds of results, with individual criteria for quality:

> findings -- well-established scientific truths --
  judged by truthfulness and rigor

> observations -- reports on actual phenomena --
  judged by interestingness

> rules-of-thumb -- generalizations, signed by an author (but perhaps not fully supported by data) --
  judged by usefulness

with freshness as criterion for all

Parnas, “Modular Structure” - Results

- ES: To remedy the situation, an improved solution is proposed, whereby modules are organized into a tree-like hierarchical structure. Also, to aid the overall development process, the idea of a software module guide is introduced.

- ES?: An enhancement is described based on the creation of a tree-structured hierarchical Module Guide which characterizes what is “hidden” by each module.

- XH: … generates a number of findings concerning modular design, software organization, and software maintenance, including the fact that applying principles of information hiding to large, complex software systems results in an explosion of software modules and the need for module guide documentation and hierarchical organization of the modules.

- (was ET, now EM): The enhanced method adds a software modularization design template in a design approach called “design through documentation”, thereby enabling the promotion of system design understanding, reduction in communication, and increase in reuse.

- ESNA: An experimental evaluation of information hiding for problems of real-world complexity has been carried out using the Operational Flight Program for the A-7E aircraft.

- ES: An enhanced process of modular definition and specification for the project for the Operational Flight Program for the A-7E aircraft is described based on modular characteristics and secret hidden was implemented. In comparison with existing solutions, it serves as a better resource and reference for developers for system construction and modification.

Composable Software Research at Carnegie Mellon

Research Objectives

Research Setting

Real World Practical problem

Idealized problem

Research product (technique, method, model, system, …)

Research Setting

Validation Task 1:
Does the product solve the idealized problem?

Real World Solution to practical problem

Idealized problem

Research Setting

Solution to idealized problem

Validation Task 2:
Does the result help to solve the practical problem?

Types of Research Validation

• Persuasion
  I thought hard about this, and I believe...

• Implementation
  Here is a prototype of a system that ...

• Evaluation
  Given these criteria, the object rates as ...

• Analysis
  Given the facts, here are consequences ...
  
  Formal model
  Rigorous derivation and proof

  Empirical model
  Data on use in controlled situation

• Experience
  Report on use in practice
  
  Qualitative model
  Narrative

  Decision criteria
  Comparison of systems in actual use

  Empirical model
  Data, usually statistical, on practice

Complete Research Result

Validation Task 1:
Does the product solve the idealized problem?

Validation Task 2:
Does the result help to solve the practical problem?

Parnas, “Modular Structure” - Validation

• ES: To demonstrate the applicability of these improvements, authors re-implement a moderately complex software system, the Operation Flight Program for the A-7E aircraft, and report limited success in the early stages.

• ET?: An ongoing real-world example is provided confirming the effectiveness of its support.

• XIII:...
  (new ET again) Examples are provided confirming the effectiveness of its support for information hiding in design.

• XII: They indicate that the use of information hiding in complex systems is met by “design through documentation”.

• ESNA: The benefit of reduced cost due to independent module design and revision holds for problems of real-world complexity if information hiding is accompanied by a module guide that specifies module responsibilities and interfaces. Limitations remaining to the application of information hiding to problems of real-world complexity include further research into the size and format of the module guide and reliance on the requirements document for division of responsibility between behavior, hardware, and software.

• ES: These improvements have not been confirmed by the tests of the running program (as of 1985 but was subsequently the project was proven to be a great success) but during development it was shown through observations that it was helpful.
“No-No” s for Software Engineering Research

- Assume that a result demonstrated for a 10K-line system will scale to a 500K-line system
- Expect everyone to do things “my way”
- Believe functional correctness is sufficient
- Assume the existence of a complete, consistent specification
- Just build things without extracting enduring lessons
- Devise a solution in ignorance of how the world really works

Building Blocks for Research

<table>
<thead>
<tr>
<th>Question</th>
<th>Strategy/Result</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feasibility</td>
<td>Qualitative model</td>
<td>Persuasion</td>
</tr>
<tr>
<td>Characterization</td>
<td>Technique</td>
<td>Implementation</td>
</tr>
<tr>
<td>Method/Means</td>
<td>System</td>
<td>Evaluation</td>
</tr>
<tr>
<td>Generalization</td>
<td>Empirical model</td>
<td>Analysis</td>
</tr>
<tr>
<td>Selection</td>
<td>Analytic model</td>
<td>Experience</td>
</tr>
</tbody>
</table>

A Common Plan

<table>
<thead>
<tr>
<th>Question</th>
<th>Strategy/Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can X be done better?</td>
<td>Build a Y</td>
</tr>
<tr>
<td>Generalization</td>
<td>Empirical model</td>
</tr>
<tr>
<td>Selection</td>
<td>Analytic model</td>
</tr>
</tbody>
</table>

A Common, but Bad, Plan

<table>
<thead>
<tr>
<th>Question</th>
<th>Strategy/Result</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can X be done better?</td>
<td>Devising a technique</td>
<td>“Look, it works!!”</td>
</tr>
<tr>
<td>Generalization</td>
<td>Empirical model</td>
<td>Analysis</td>
</tr>
<tr>
<td>Selection</td>
<td>Analytic model</td>
<td>Experience</td>
</tr>
</tbody>
</table>
Two Other Good Plans

<table>
<thead>
<tr>
<th>Question</th>
<th>Strategy/Result</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can X be done at all?</td>
<td>Qualitative model</td>
<td>“Look, it works!!”</td>
</tr>
<tr>
<td>Characterization</td>
<td>Technique</td>
<td>Implementation</td>
</tr>
<tr>
<td>Method/Means</td>
<td>Build a Y that does X</td>
<td>Evaluation</td>
</tr>
<tr>
<td>Is X always true of Y?</td>
<td>Empirical model</td>
<td>Check proof</td>
</tr>
<tr>
<td>Selection</td>
<td>Formally model Y, prove X</td>
<td>Experience</td>
</tr>
</tbody>
</table>

Sometimes a breakthrough (but sometimes nonsense)

<table>
<thead>
<tr>
<th>Question</th>
<th>Strategy/Result</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feasibility</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Persuasion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change basic assumptions</td>
<td>Technique</td>
<td>Implementation</td>
</tr>
<tr>
<td>Method/Means</td>
<td>System</td>
<td>Evaluation</td>
</tr>
<tr>
<td>Generalization</td>
<td>Empirical model</td>
<td>Analysis</td>
</tr>
<tr>
<td>Selection</td>
<td>Analytic model</td>
<td>Experience</td>
</tr>
</tbody>
</table>

Problem Types, Products, Validations

24 classic SE papers, 10 ICSE 2001 software architecture papers

Executive Summary

- Technical ideas in software engineering evolve in predictable ways
- This evolution pattern sets expectations for software architecture research
- Lessons for developers: what to expect of software architecture research, now and in the future
- Lessons for researchers: useful research paradigms and validation techniques
  > I offer these as observations, in Brooks’ sense
Pro Forma Abstracts

• Idea: Identify the methodological “signature” of the research and state major points in a predictable way
  > Existing model-type models are deficient in dealing with properties of solution strategy.
  > An enhanced model-type is described, capable of providing more accurate analyses/predictions of properties in solution strategy designs.
  > The model has been tested by comparing analyses/predictions with empirically measured values of properties.

Pro Forma Abstracts

• Idea: Identify the methodological “signature” of the research and state major points in a predictable way
  > Existing state models are deficient in dealing with dependability of object-oriented designs.
  > An enhanced state model is described, capable of providing more accurate analyses/predictions of dependability in object-oriented designs.
  > The model has been tested by comparing analyses/predictions with empirically measured values of dependability.

• Risk: Filling in slots mechanically and missing major points.
• Suggestion: In addition to Newman’s skeleton, add a sentence describing the essence of the result:
  > (add between 2 & 3) This enhanced model adds probabilities of state transitions, thereby enabling Markov analysis of failure.

Newman’s Pro Forma Templates for Engineering

EM: Enhanced model
Existing model-type models are deficient in dealing with properties of solution strategy. An enhanced model-type is described, capable of providing more accurate analyses / predictions of properties in solution strategy designs. The model has been tested by comparing analyses / predictions with empirically measured values of properties.

ES: Enhanced solution
Studies of existing artifact-type have shown deficiencies on property. An enhanced design for an artifact-type is described, based on solution strategy. In comparison with existing solutions, it offers enhanced levels of property, according to analyses based on model-type. These improvements have been confirmed / demonstrated in tests of a working artifact-type based on the design.

ET: Enhanced tool
The effectiveness of model-type / solution strategy in supporting the design of artifact-type has been demonstrated. An enhanced tool / method is described for the design of artifact-type based on model-type / solution strategy. Examples are provided confirming the effectiveness of its support for model-type / solution strategy.

Additional Pro Forma Templates for HCI

RS: Radical solution
A radical solution to the problem of problem definition is described, based on solution strategy. In comparison with existing normal solutions it offers advantages, which have been demonstrated in preliminary tests, but it leaves a number of side effects to be addressed including list of side effects. Strategies are suggested for addressing these side effects.

XH: Experience and/or Heuristic
Studies reported here of application supported by supporting technology generate a number of findings concerning issues, including list-of-findings. They indicate that requirement is / is not met by design-heuristic.

ESNA: Existing Solution Applied to New Area [Latronico]
The technique existing technique has been successfully applied in area-x with benefits benefits for area-x. It is hoped that technique will provide similar benefits in area-y. An experimental evaluation of technique for area-y has been carried out using system from area-x. Benefits benefits for area-y have been shown to hold (optional: if technique is altered by alteration list). Limitations remaining to the application of technique to area-y include limitation list.
Examination of real-life complex software projects has shown that information hiding principles have not been utilized by industrial software development teams. Authors have analyzed the possible reasons for lack of such adoption, and conclude that although applicable to small problems, the modularization of a software project based on information hiding does not work well with large complex software, where the number of modules is in the hundreds. To remedy the situation, an improved solution is proposed, whereby modules are organized into a tree-like hierarchical structure. Also, to aid the overall development process, the idea of a software module guide is introduced. To demonstrate the applicability of these improvements, authors re-implement a moderately complex software system, the Operation Flight Program for the A-7E aircraft, and report limited success in the early stages.

The effectiveness of information hiding in supporting the design of maintainable software systems has been argued. An enhancement is described based on the creation of a tree-structured hierarchical Module Guide which characterizes what is "hidden" by each module. An ongoing real-world example is provided confirming the effectiveness of its support.

The study reported here of modularization of a complex software system supported by information hiding generates a number of findings concerning modular design, software organization, and software maintenance, including the fact that applying principles of information hiding to large, complex software systems results in an explosion of software modules and the need for module guide documentation and hierarchical organization of the modules.

The effectiveness of information hiding in supporting the design of software systems has been demonstrated. An enhanced method is described for the design of software systems based on the principle of information hiding to decompose a project into work assignments or modules. The enhanced method adds a software modularization design template in a design approach called "design through documentation", thereby enabling the promotion of system design understanding, reduction in communication, and increase in reuse. Examples are provided confirming the effectiveness of its support for information hiding in design.
Parnas Modular Structures PFA#5

- XH
  Studies reported here of designing software systems supported by the principle of information hiding generate a number of findings concerning the feasibility of applying the concepts to realistic problems, including the deficiency of examples examined in conference paper and the lack of a model for others to follow. They indicate that the use of information hiding in complex systems is met by "design through documentation".

Parnas Modular Structures PFA#6

- ESNA
  The technique of information hiding has been successfully applied to multiple small and straightforward academic software design problems, providing the major benefit of reduced cost by "allowing modules to be designed and revised independently." It is hoped that information hiding will provide similar benefits to problems of real-world complexity. An experimental evaluation of information hiding for problems of real-world complexity has been carried out using the Operational Flight Program for the A-7E aircraft. The benefit of reduced cost due to independent module design and revision holds for problems of real-world complexity if information hiding is accompanied by a module guide that specifies module responsibilities and interfaces. Limitations remaining to the application of information hiding to problems of real-world complexity include further research into the size and format of the module guide and reliance on the requirements document for division of responsibility between behavior, hardware, and software.

Parnas Modular Structures PFA#7

- ES
  Studies of the use of information hiding through modularization have shown deficiencies in its usability in large-scale systems (of over 25 modules) because of information overload and oversight effect in the event of changes. An enhanced process of modular definition and specification for the project for the Operational Flight Program for the A-7E aircraft is described based on modular characteristics and secret hidden was implemented. In comparison with existing solutions, it serves as a better resource and reference for developers for system construction and modification. Theses improvements have not been confirmed by the tests of the running program (as of 1985 but was subsequently the project was proven to be a great success) but during development it was shown through observations that it was helpful.