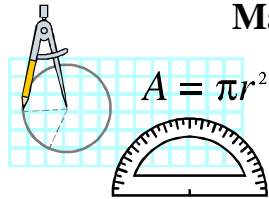


# Dissertation Defense



## Using Object-Oriented Design Complexity Metrics to Predict Maintenance Performance



Dissertation Defense  
by  
Rajendra K. Bandi

© Rajendra K Bandi

Slide 1



## *Presentation Outline*

- ✦ Introduction
- ✦ Background & Motivation
- ✦ Software Metrics: What & Why?
- ✦ What to measure and how?
- ✦ Research Question
- ✦ Research Methodology
- ✦ Results
- ✦ Significance of the Research
- ✦ Future Research

© Rajendra K Bandi

Slide 2

# Dissertation Defense




## *Introduction*

**The objective of this research is to  
Validate Metrics derived at the design stage  
in object-oriented software development**

© Rajendra K Bandi

Slide 3



## *Background & Motivation*

- ❖ **OO Paradigm becoming increasingly popular**
- ❖ **Claimed advantages [Booch, 1995; Cox, 1986]**
  - ❖ Easier Maintenance, Greater Reuse, Improved Quality
  - ❖ Improved Productivity, Economic gains
- ❖ **OO Reality**
  - ❖ Some evidence seen to support claimed advantages [Mancl & Havanas, 1990]
  - ❖ But, the reverse also has been seen [Anderson & Sheffler, 1992]
  - ❖ Maintenance burden will not completely disappear [Wilde & Huitt, 1992]
- ❖ **Industry sponsored research opportunity**
  - ❖ COMSOFT, Handbooks

© Rajendra K Bandi

Slide 4

# Dissertation Defense

## *Acceptance of OO Paradigm*

### ⊗ Need to address

❖ NOT just the technical requirements

❖ BUT also management needs to

- ◆ plan,
- ◆ monitor,
- ◆ control, &
- ◆ deliver cost effective software systems

### ⊗ Renewed interest in software metrics

❖ potential for use in procedures to control software development process

## *Software Measurement/Metrics*

### ⊗ What is Measurement

❖ the process of assigning symbols, usually numbers, to represent an attribute of the entity of interest, by rule [Shepperd, 1995]

### ⊗ Why Measure

❖ “*You Cannot Control What You Cannot Measure*” [DeMarco, 1982]

❖ “*When you can measure what you are speaking about & express it in numbers you know something about it; but when you can't measure, your knowledge is of a meager & unsatisfactory kind*” [Lord Kelvin]

❖ objective evaluation of a design/program

- ◆ good design or “sheep dressed as lamb”

❖ quantitative evaluation of the value of technology, for example, in terms of productivity, effort, time, and maintainability

❖ basis for good cost estimation

# Dissertation Defense

## *What to measure?*

- ❖ **Design Phase is where the Signature of a System is Created. Design is like a skeleton onto which flesh is put at later stages**
- ❖ **The Major Professional Challenge is to Conquer the Complexity of the System [Parikh, 1983]**
- ❖ **Measuring Design complexity very popular with software metrics researchers**
- ❖ **Design complexity shown to correlate well with**
  - ❖ system development effort
  - ❖ maintainability
  - ❖ defect rate etc.

© Rajendra K Bandi

Slide 7

## *Complexity?*

- ❖ **Complex:**
  - ❖ Whole made up of complicated or interrelated parts
  - ❖ a group of obviously related units of which the degree & nature of the relationship is imperfectly known [Webster's dictionary]
  - ❖ numerosity of its composition (large number of properties) [Bunge, 1977]

© Rajendra K Bandi

Slide 8

# Dissertation Defense

## *How to measure Design Complexity?*

- ❖ Several complexity metrics exist for the traditional systems (software science, cyclomatic complexity etc.) [Halstead, 1977; McCabe, 1976]
- ❖ **Advantages**
  - ❖ Very well known. Several studies have been conducted.
  - ❖ Validated as predictors of development effort, maintainability etc.
- ❖ **Disadvantages**
  - ❖ Not appropriate for OO systems. Do not capture the unique features of OO systems. [Chidamber & Kemerer, 1994; Keyes, 1992; Taylor 1993 etc.,]
  - ❖ Not theoretically sound [Weyuker, 1988]

© Rajendra K Bandi

Slide 9

## *How to measure Design Complexity? (continued)*

- ❖ Several more new metrics being proposed for OO systems [Chidamber & Kemerer, 1994; Abbott, 1993; Chen & Lu, 1993; Abreu, 1995 etc.]
- ❖ **Advantages**
  - ❖ Appropriate for OO systems
- ❖ **Disadvantages**
  - ❖ Metrics criticized for ambiguity in definitions
  - ❖ Most metrics have not yet been validated
  - ❖ Just proposed or only subjective validation

© Rajendra K Bandi

Slide 10

# Dissertation Defense

## *So what is the choice?*

- ❖ **choose well known but inappropriate metrics**
- ❖ **chose appropriate but not yet validated metrics**

© Rajendra K Bandi

Slide 11

## *Validating metrics*

- ❖ **Metrics should be theoretically sound**
- ❖ **Should be a predictor of some quality of interest [Fenton, 1991]**

---

**Most metrics in traditional systems have been validated by showing the metrics' ability to predict maintainability**

- ❖ **The World of Software Maintenance is a World in Which Programmers Spend Half of All Their Lifetime [Parikh, 1983]**
- ❖ **US corporations spend more than \$100 billion annually on Software Maintenance [Moad, 1990]**
- ❖ **Maintenance burden will not completely disappear in OO systems [Wilde & Huitt, 1992]**

© Rajendra K Bandi

Slide 12

# Dissertation Defense



## ***Research Problem***

- ❖ *Can we have complexity metrics defined to measure the unique aspects of the OO design, which not only meet the theoretical rigor for good metrics, but also are indicative of some quality like maintainability?*

© Rajendra K Bandi

Slide 13



## ***Research Approach***

- ❖ **Analytical Evaluation of Design Complexity Metrics**
- ❖ **Empirical Evaluation of Design Complexity Metrics**
  - ❖ **Controlled laboratory experiment**

© Rajendra K Bandi

Slide 14

# Dissertation Defense

## *OO Design Complexity Metrics (Focus Metrics)*

- ❖ **Interface Size (IS)** [Abbott, 1993]
- ❖ **Interaction Level (IL)** [Abbott, 1993]
- ❖ **Operation Argument Complexity (OAC)** [Chen and Lu, 1993]
- ❖ **Attribute Complexity (AC)** [Chen and Lu, 1993]

© Rajendra K Bandi

Slide 15

## *Example*

### Class Quadrilateral

#### Attributes:

float x1, y1;  
float x2, y2;  
float x3, y3;  
float x4, y4;

#### Methods:

Boolean hasVertex(float x, float y)

Number of parameters = 3 ( 2 float, 1 boolean)

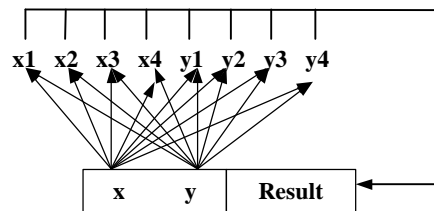
IS for float = 2; IS for boolean = 0

**IS** = Number of parameters +  $\Sigma$  IS of parameters = 7

**IL** = Number of interactions +  $\Sigma$  strength of interactions = 24 + (16\*4) = 88

**OAC** =  $\Sigma$  parameter size = 2 + 2 + 0 = 4

**AC** =  $\Sigma$  Data Attribute size = 2\* 8 = 16



© Rajendra K Bandi

Slide 16

# Dissertation Defense

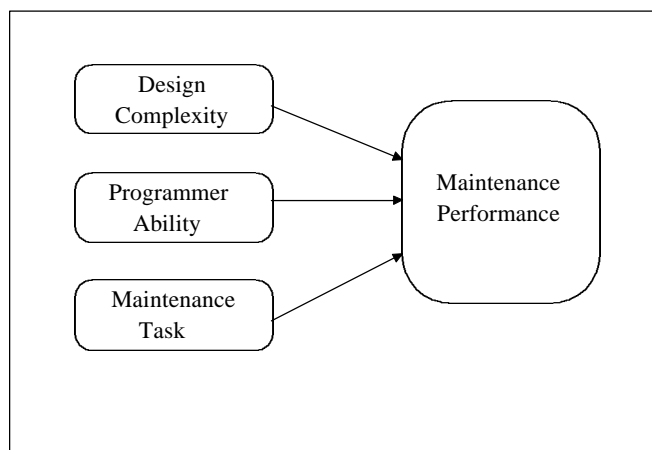
## *Analytical Evaluation - Weyuker's Properties*

- ❖ Non-Coarseness
- ❖ Granularity
- ❖ Non-uniqueness
- ❖ Design Details are important
- ❖ Monotonicity
- ❖ Non-equivalence of Interaction
- ❖ Permutation changes Complexity
  - ❖ not relevant for OO systems
- ❖ Renaming Property
- ❖ Interaction Increases Complexity
  - ❖ not relevant for OO systems

© Rajendra K Bandi

Slide 17

## *Empirical Evaluation - Research Model*



© Rajendra K Bandi

Slide 18

# Dissertation Defense



## ***Research Propositions*** ***(derived from the research model)***

**Our Main focus (to validate metrics)**


**P1: Increased system complexity leads to increased time to make changes**

**Secondary objective**

**P2: Increased programmer's ability leads to decreased time to make changes**

© Rajendra K Bandi

Slide 19



## ***Hypotheses (derived from P1)***

**H1: Increased Interaction Level of a class leads to increased time to make changes**

**H2: Increased Interface Size of a class leads to increased time to make changes**

**H3: Increased Operation Argument Complexity of a class leads to increased time to make changes**

**H4: Increased Attribute Complexity of a class leads to increased time to make changes**

© Rajendra K Bandi

Slide 20

# Dissertation Defense

## *Hypotheses (derived from P2)*

- H5: Increased number of IS courses taken leads to decreased time to make changes**
- H6: Better academic performance in the courses leads to decreased time to make changes**
- H7: Increased duration of programming experience leads to decreased time to make changes**
- H8: Increased duration of OO experience leads to decreased time to make changes**

© Rajendra K Bandi

Slide 21

## *Research Design*

- ❖ **Key Independent Variables (Complexity)**
  - ❖ Interaction Level
  - ❖ Interface Size
  - ❖ Operation Argument Complexity
  - ❖ Attribute Complexity
- ❖ **Other Independent Variables**
  - ❖ Programmer ability
    - ◆ No. of IS course credits taken
    - ◆ Academic Performance
    - ◆ Total programming experience
    - ◆ OO experience
- ❖ **Dependent Variable (Maintenance Performance)**
  - ❖ Maintenance Time ( time taken make changes)

© Rajendra K Bandi

Slide 22

# Dissertation Defense

## *The experiment*

- ❖ **Controlled laboratory experiment**
- ❖ **Students of Advanced C++ class (93 participants)**
- ❖ **Two Treatments**
  - ❖ 1 perfective maintenance (adding new functionality)
  - ❖ 1 corrective maintenance (changing existing functionality)
- ❖ **For each treatment**
  - ❖ two versions of design created
  - ❖ a low complexity version and a high complexity version
- ❖ **Multiple sections with**
  - ❖ different instructors
  - ❖ but, same teaching material, assignments
- ❖ **Compulsory assignment**

© Rajendra K Bandi

Slide 23

## *Test Results: Summary*

- ❖ **ANOVA**
  - ❖ System classified as high or low complexity based on metrics values
  - ❖ As expected, system classified as high complexity takes significantly (statistically) more time than a system classified as low complexity
  - ❖ No significant differences in times across instructors or quarters
- ❖ **Multiple Regression Analysis**
  - ❖ Complexity as indicator variable, and programmer variables
  - ❖ Complexity still statistically significant
- ❖ **What does it mean?**
  - ❖ Complexity significant even after controlling for programmer ability
  - ❖ The instructor or the quarter in which the study is done have not influenced the observed time
  - ❖ *The metrics are valid measures of complexity*

© Rajendra K Bandi

Slide 24

# Dissertation Defense

## *Test Results: Summary*

- ❖ **Simple Regression Analysis: Metrics Vs. Time**
  - ❖ Each metric found to be significant
  - ❖ Able to reject Null Hypotheses for H1 .. H4
- ❖ **Multiple Regression Analysis: Metrics Vs. Time**
  - ❖ Multicollinearity detected among IL, IS, OAC
  - ❖ Using AC plus any one of (IL, IS, OAC) seems to be the most useful strategy

© Rajendra K Bandi

Slide 25

## *Test Results: Summary*

- ❖ **Regression Analysis (simple & multiple)  
Programmer variables Vs. Time**
  - ❖ No statistical significance
  - ❖ Fail to reject Null Hypotheses for H5 .. H8
- ❖ **Direction of regression coefficients**
  - ❖ No. of Credits -- inconsistent
  - ❖ Academic Performance -- Negative
  - ❖ Total Experience -- Positive
  - ❖ OO Experience -- Negative
  - ❖ Non OO Experience -- Positive

© Rajendra K Bandi

Slide 26

# Dissertation Defense

## *Conclusions & Limitations*

### ❖ **Conclusions**

- ❖ IL, IS, OAC, & AC are valid OO design complexity metrics for predicting maintenance
- ❖ No need to measure all the metrics values
- ❖ No conclusive results on programmer variables
  - ◆ non OO experience seems to hinder programmer ability

### ❖ **Limitations**

- ❖ Limited scope of systems used
- ❖ Limited experience of participants in OO systems

© Rajendra K Bandi

Slide 27

## *Future Research*

### ❖ **Conduct a study to separately capture time**

- ❖ to understand
- ❖ make changes
- ❖ test changes

### ❖ **Study with more OO experienced participants**

### ❖ **Conduct a longitudinal study**

- ❖ investigate metrics ability to explain reliability, system deterioration, frequency of changes, etc.

### ❖ **Design-based Vs. Analysis based metrics?**

### ❖ **Pedagogical perspective**

- ❖ usefulness of focusing on software maintenance in programming classes

© Rajendra K Bandi

Slide 28

# Dissertation Defense



## *OO Design Complexity Metrics Definitions*

### ✧ **Interface Size (IS)**

gives a measure of the means for information flow

$$IS = K3*(\text{No. of parameters}) + K4*(\text{Sum of parameter sizes})$$

### ✧ **Interaction Level (IL)**

specifies the amount of interaction that can occur

$$IL = K1*(\text{No. of interactions}) + K2*(\text{Strength of interactions})$$

© Rajendra K Bandi

Slide 29



## *OO Design Complexity Metrics Definitions*

### ✧ **Operation Argument Complexity (OAC)**

measures the complexity of the method parameters

$$OAC = \sum P(i)$$

where, P(i) is the value of each argument in each operation of the class

### ✧ **Attribute Complexity (AC)**

measures the complexity of the attributes of the class

$$AC = \sum R(i)$$

where, R(i) is the value of each attribute in the class

© Rajendra K Bandi

Slide 30

# Dissertation Defense

## *Literature Review - Studies involving OO Systems*

- ❖ **Chidamber & Kemerer, 1991, 1994**
  - ❖ Present theoretical work that build a suite of class-based metrics for OO design
- ❖ **Tegarden and Sheetz, 1992, 1993, 1995**
  - ❖ Investigate Software Science, Cyclomatic Complexity as possible indicators of complexity for OO systems
- ❖ **Abreu and others 1994, 1995, 1996**
  - ❖ Propose a suite of system-based metrics for OO systems
- ❖ **Banker et al, 1991**
  - ❖ Propose a new metric 'object points' similar to function points

© Rajendra K Bandi

Slide 31

## *Literature Review - Studies involving OO Systems contd...*

- ❖ **Thomas, 1989**
  - ❖ Suggests that LOC can be used as a metric of reuse in OO systems
- ❖ **Abbott, 1993**
  - ❖ Proposes Interaction Level and Interface Size metrics
- ❖ **Chen & Lu, 1993**
  - ❖ Propose Operation Argument Complexity and Attribute Complexity metrics

© Rajendra K Bandi

Slide 32

# Dissertation Defense

## *Literature Review - Complexity Metrics & Maintenance*

### ❖ **Rombach, 1987**

- ❖ Design complexity metrics predict maintenance almost as well as the code measures (controlled experiment)

### ❖ **Mancl & Havanas, 1990**

- ❖ Evaluate maintenance differences between OO and conventional programming (case study)

### ❖ **Li & Henry, 1993**

- ❖ OO complexity metrics can be used as predictors of maintenance effort (field study)