

# Innovations in Software Engineering Education

*School of Systems &  
Enterprises*

*Larry Bernstein*



# Vision

- Reliable Software Products produced within budget and on-time.
- Metrics based Software Management
- Quantitative Software Design
- Understanding Technical Foundations
- Repeatable Processes
- State-of-the-Art

# Be able to answer these Four Questions

- Is this proposed software centric system feasible?
- If it is, how much will it cost?
- If we are willing to pay, how long will it take to build and to deploy?
- What is the development plan, especially the detailed schedule?

# Themes across all Courses

- System Focus
- Quantitative Analysis
- Trustworthiness
- Case Histories (tailored to application domains)
- Parnas & Boehm Papers

# Typical Course Sequence

SyS 625	Introduction to Systems & Software Engineering
SSW 540	Survey of Systems & Software Engineering
SSW 533	Cost Estimation & Metrics
SSW 565	Architecture & Component Design
SSW 567	Testing, QA & Maintenance
SSW 689	Software Reliability
Four Electives	Courses selected from Sys, EM, CS, or CpE programs with advisor approval of study plan.

# Vision

- Reliable Software Products produced within budget and on-time.
- Metrics based Software Management
- Quantitative Software Design
- Understanding Technical Foundations
- Repeatable Processes
- State-of-the-Art

# Be able to answer these Four Questions

- Is this proposed software centric system feasible?
- If it is, how much will it cost?
- If we are willing to pay, how long will it take to build and to deploy?
- What is the development plan, especially the detailed schedule?

# Innovations

- Live-Thru Case Histories
- Lambda Protocol
- Integration with Systems Engineering
- Reliability focused Software Engineering
- Model Driven Design & Testing
- Design Simplification

# Elements of Live-Through

- Short 4-week project
- Inexperienced students
- Classic Traps
- No programmed guidance; help provided as requested.
- Simple Development or Maintenance Project

# Live-Through Case: Regression Testing and Creeping Featurism

- Given a working CRM
- Add a Warehouse Message
- Add Postal Code to address
  - Define Code
- Product presentation: week 4
- New CRM version: end of week 2
- Credit check interface feature emerges: week 3

# Model Clash

- Did you add features/functionality to impress the customer?
- What makes you think they will impress?
- Postal Code: Drop Down or fill in?

## Model clash:

”I should be able to learn entire system in an hour,” service rep

“ I need to keep implementation simple,” developer

# Lambda Protocol

- Prospectus
- Measurable Operational Value
- Prototyping or Modeling
- QFD
- Schedule, Staffing, Quality Estimates
- ICED-T
- Trade-off Analysis

# Universal Software Engineering Equation

$$\textit{Reliability} (t) = e^{-k \lambda t}$$

when the error rate is constant and where  $k$  is a normalizing constant for a software shop and

$\lambda$  = Complexity/ effectiveness x staffing

# Case Study: SchedulerPro Prospectus

User friendly, efficient interface for students to create and modify class schedules.

## Features:

- Visual schedule creation and editing
- Schedule suggestion
- Schedule comparison view
- Monitor closed-out sections

# SchedulerPro Prototype Screen

The screenshot shows a software interface for scheduling classes. On the left, a sidebar lists the classes for Fall 2004:

- CS 551 - Software Engineering and Practice I
  - Professor: Bernstein L.
  - Room: E222
  - TR: 3:30-5:00 PM
  - Prereq: CS 385 or CS 590
  - Call Number: 10225
- CS 600 - Algorithms
- CS 488 - Computer Architecture
- HPL 450 - International Ethics
- PE 200 - Bowling

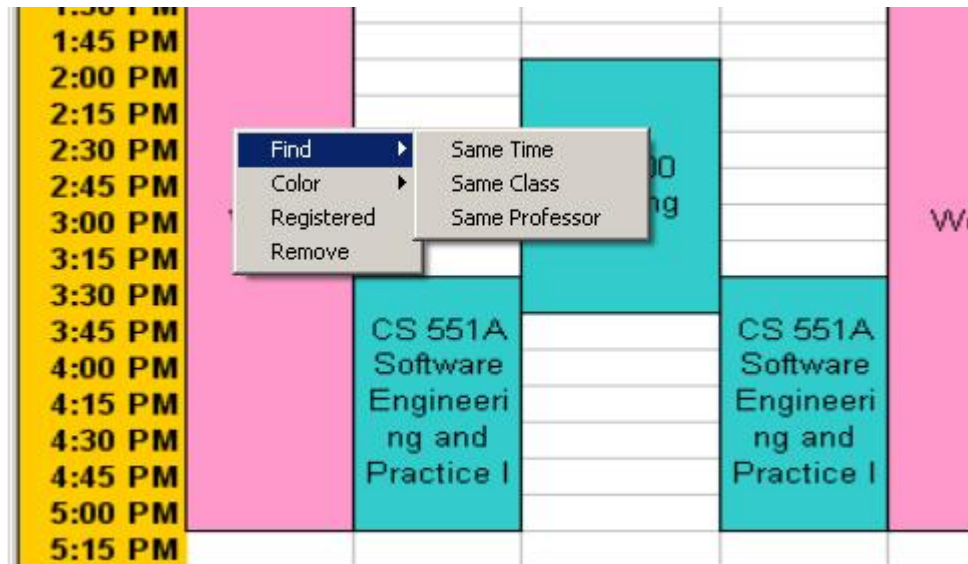
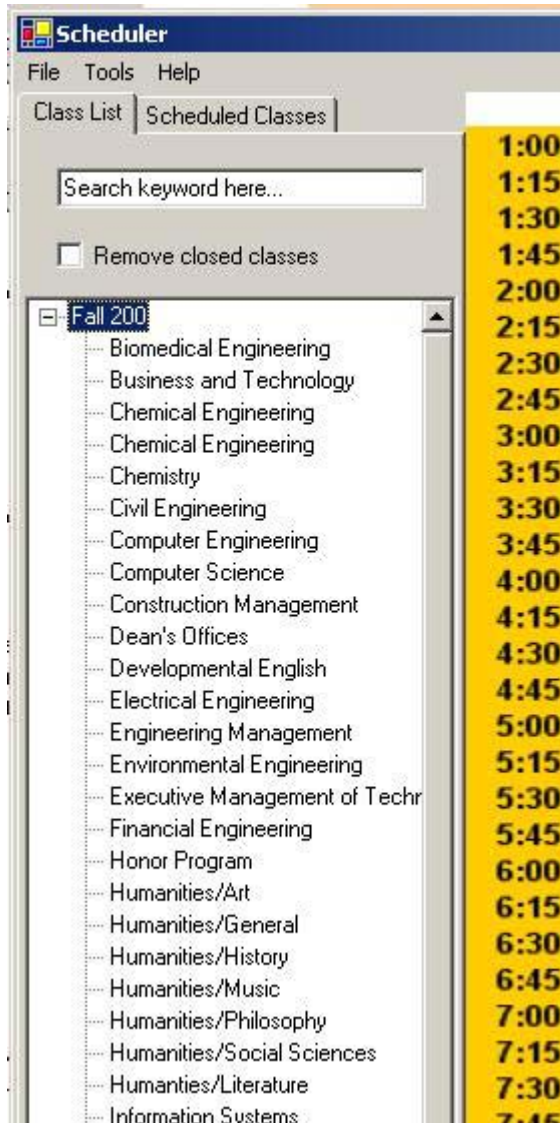
The main grid shows the following schedule:

Time	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
1:00 PM	Work				Work		
1:15 PM							
1:30 PM							
1:45 PM							
2:00 PM				PE 200 Bowling			
2:15 PM							
2:30 PM							
2:45 PM							
3:00 PM							
3:15 PM							
3:30 PM		CS 551A Software Engineering and Practice I		CS 551A Software Engineering and Practice I			
3:45 PM							
4:00 PM							
4:15 PM							
4:30 PM							
4:45 PM							
5:00 PM							
5:15 PM							
5:30 PM							
5:45 PM							
6:00 PM							
6:15 PM							
6:30 PM							
6:45 PM							
7:00 PM							
7:15 PM	CS 600 Algorithms		CS 488 Computer Architecture	HPL International Ethics			
7:30 PM							
7:45 PM							
8:00 PM							
8:15 PM							
8:30 PM							
8:45 PM							

At the bottom, a table provides details for the first two classes:

Class	Professor	Call Number
HUM 501WS - Found. Technical Communications	Mills B.	11428
HUM 503WS - Adv. Documentation Techniques	Mills B.	11436

# SchedulerPro Prototype Screen



# SchedulerPro Email Notification

**From:** schedulerpro@stevens.edu  
**Sent:** Tuesday, April 19, 2005 12:15 PM  
**To:** gdeangel@stevens.edu  
**Subject:** Notification From Scheduler Pro



This is an automated notification from Scheduler Pro. The following class is available for registration:

#### Notification Class Details

Title: Microprocessor Sys. Lab

Section: CS391C  
Call Number: 10239  
Instructor: STAFF  
Scheduled Meetings: Thursday  
Time: 11:00 AM-1:50 PM

Section: CS391D  
Call Number: 10240  
Instructor: STAFF  
Scheduled Meetings: Thursday  
Time: 2:00 PM-4:50 PM

Section: CS391A  
Call Number: 10237  
Instructor: STAFF  
Scheduled Meetings: Tuesday  
Time: 2:00 PM-4:50 PM

Section: CS391B  
Call Number: 10238  
Instructor: STAFF  
Scheduled Meetings: Wednesday  
Time: 10:00 AM-12:50 PM

This address is not monitored so please do not respond to this message. To discontinue this notification or to manage your schedule please visit the [Scheduler Pro Homepage](#).

# Measurable Operational Value SchedulerPro MOV

Reduce early semester course churn by 30%

# SchedulerPro Features

## Schedule Classes and Personal Time

- Searching
- Course Placement
- Course Detail Viewing
- Course Removal
- Scheduling Personal Blocks
- Notification (optional)
- Course Suggestions (optional)

# SchedulerPro Emerging Features

- Search available classes by:
  - ✓ Same professor
  - ✓ Similar time
  - ✓ Same or equivalent class but different sections
- Register and track registrations
  - Model Clash

# Function Point History

<b>Date</b>	<b>AFP</b>	<b>Project Length*</b>	<b>Projected Finish*</b>
January 27	141	19.7 staff months	August 2006
February 24	104	14.4 staff months	March 2006
April 17	82	8.5 staff months	May 2006

\*Using COCOMO Model

# SchedulerPro Creeping Features

Are the System Admins important stakeholders?

You bet...

# Observations from 28 Schools

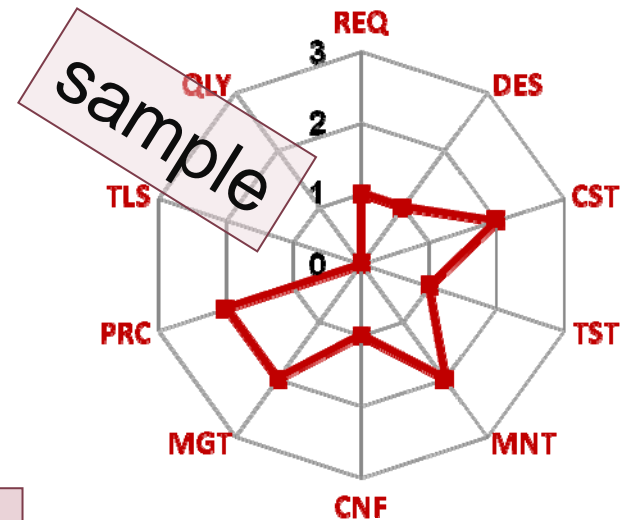
1. SWE is largely viewed as a specialization of Computer Science - much as systems engineering was often viewed as specialization of industrial engineering or operations research years ago
2. Faculty size is small - few dedicated SWE professors, making programs relatively *brittle*\*
3. Student enrollments are small
4. Many programs specialize to specific markets such as defense systems or safety critical systems
5. The target student population varies widely - anyone with Bachelors and B average to someone with CS degree and 2+ years of experience
6. Online course delivery is popular

\*A program is *brittle* if a small change in faculty composition can have a large impact on program content or capacity.

7. Goals vary – educating developers, researchers or managers
8. Wide variation in depth and breadth of SWEBOK coverage Many programs have required or semi-required courses that cover material that is either not in the SWEBOK at all or is not emphasized in the SWEBOK
9. Some significant topics are rarely mentioned - agility, software engineering economics, systems engineering
10. Some topics are ubiquitous - formal methods and architecture
11. “Object-oriented” is the standard development paradigm - creating a “clash” with many systems engineering programs that emphasize structured or procedural methods

# SWEBOK Coverage in Introductory Course

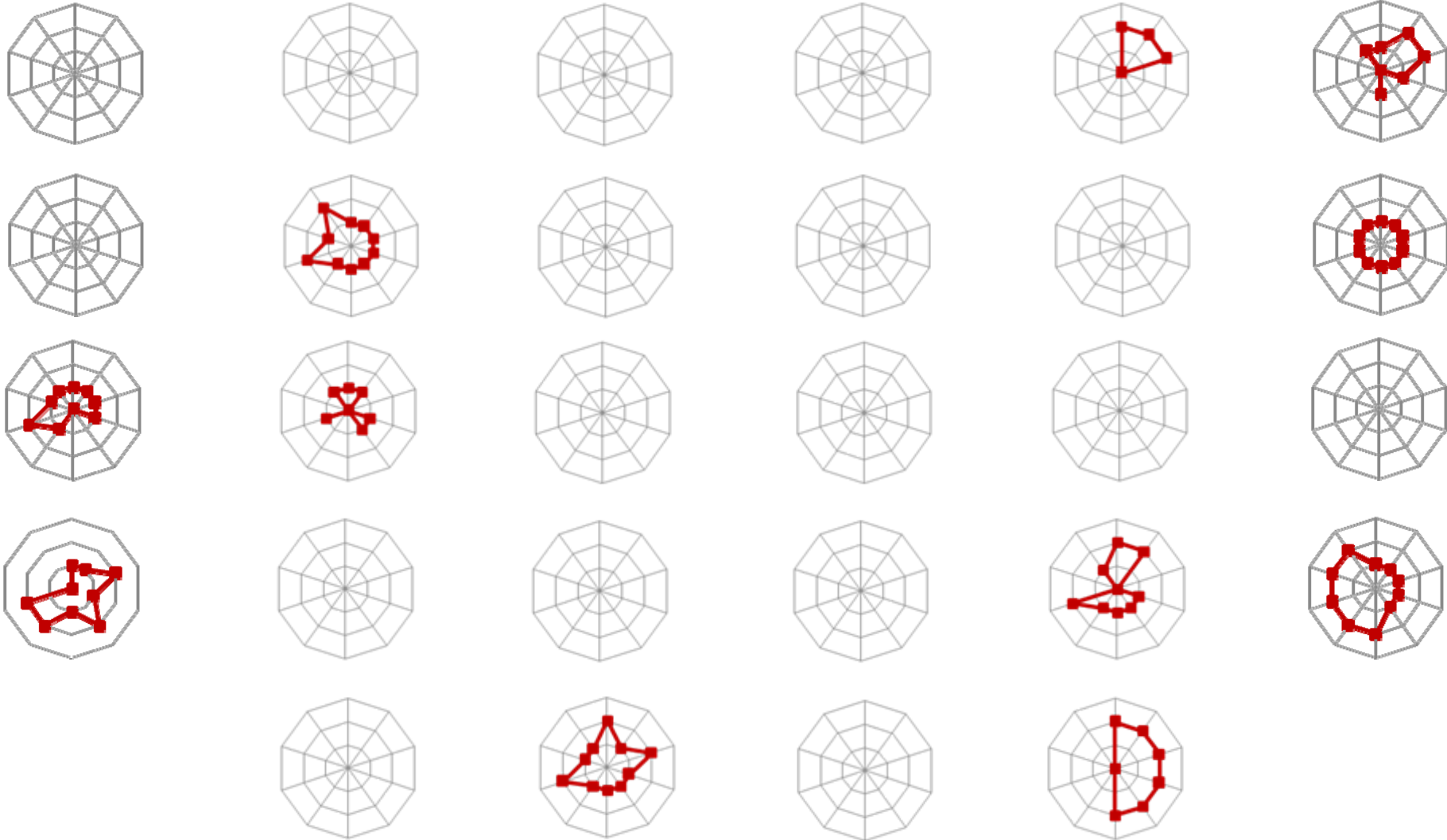
- 0: No coverage of topic
- 1: Some coverage of topic
- 2: Significant coverage of topic
- 3: Complete Coverage



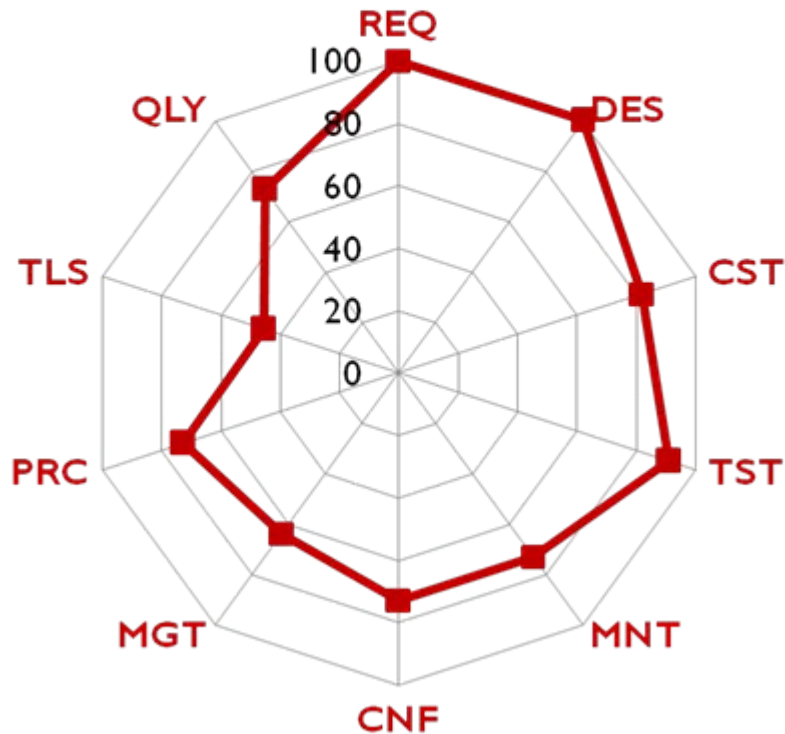
- ✿ Only 39% of the programs offer an introductory course
- ✿ Most common reason for not offering course: expect students to have industrial experience before entering program

<b>REQ</b>	Requirements
<b>DES</b>	Design
<b>CST</b>	Construction
<b>TST</b>	Testing
<b>MNT</b>	Maintenance
<b>CNF</b>	Config. Mgmt.
<b>MGT</b>	Management
<b>PRC</b>	Process
<b>TLS</b>	Tools and Methods
<b>QLY</b>	Quality

# Individual SWEBOK Coverage in Introductory Course



# Composite SWEBOK Coverage in Introductory Courses

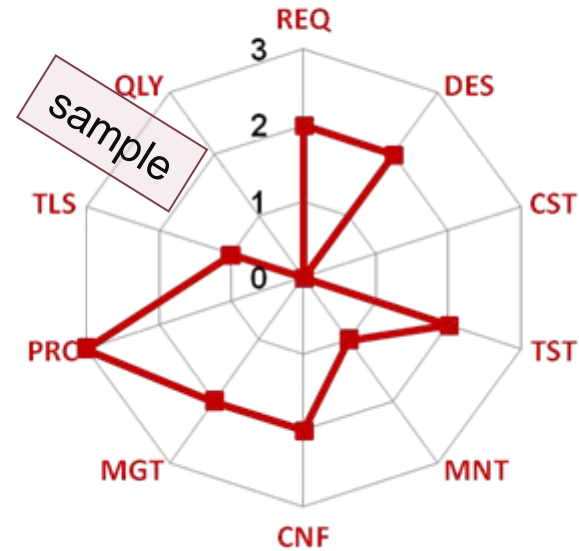


- Reasonable consistency in coverage across programs that offer an introductory course
- Requirements and design are best covered
- Tools and quality are least covered

**% of Programs with coverage**

# SWEBOK Coverage: Required & Semi-Required Courses

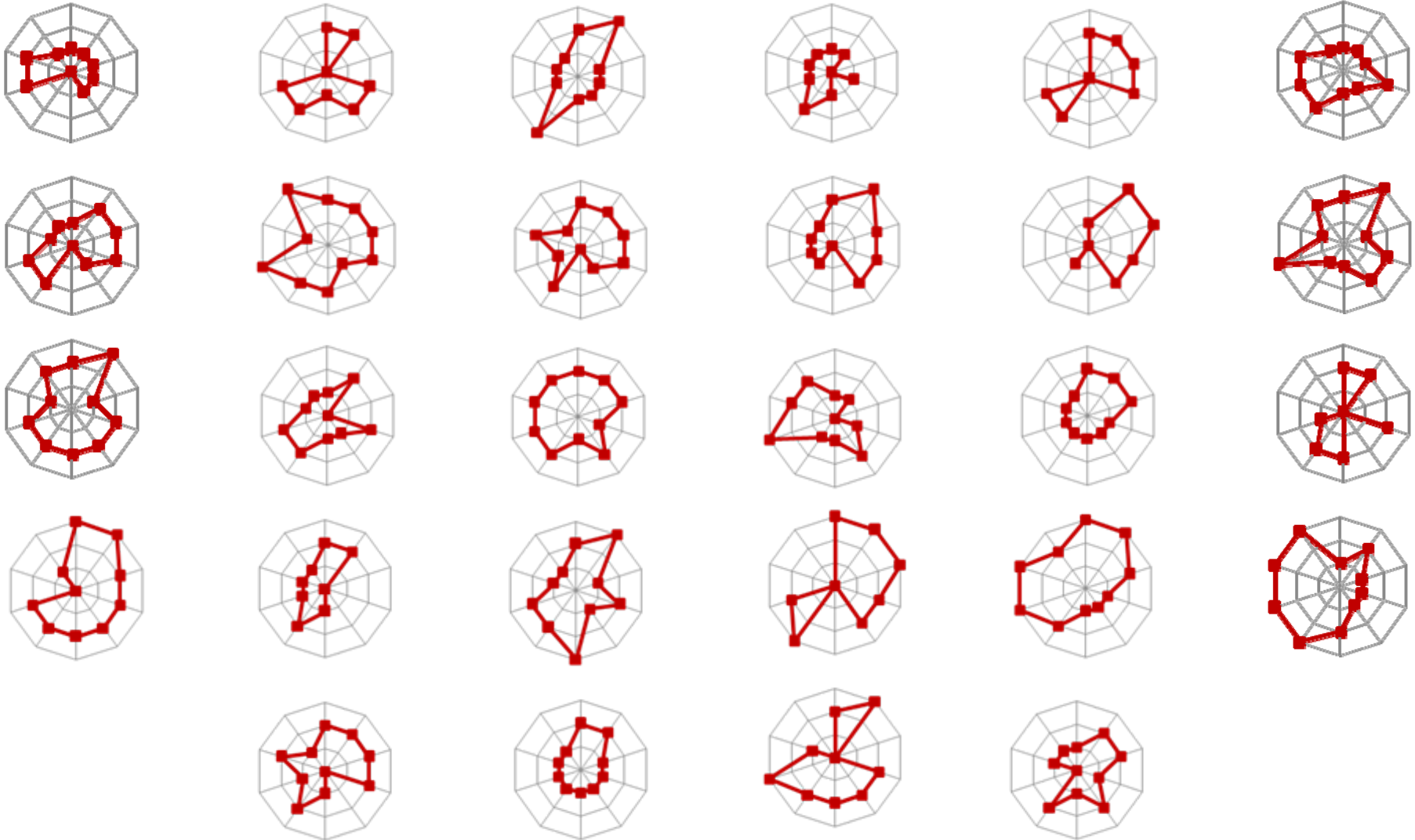
- 0: No coverage of topic
- 1: Some coverage but no dedicated course
- 2: One dedicated course
- 3: Two or more dedicated courses



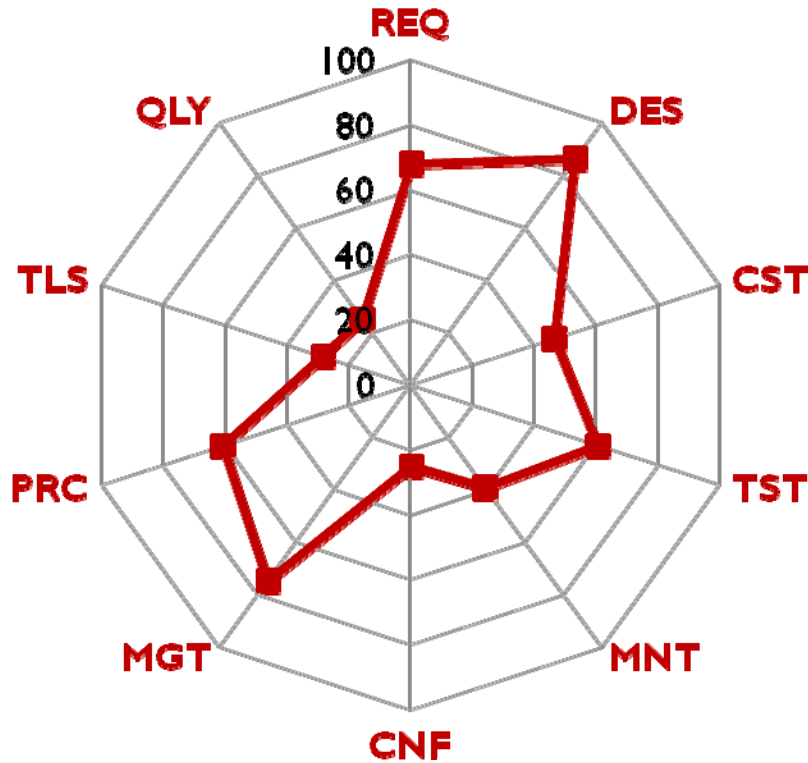
- ✿ **Required:** Student must take the course
- ✿ **Semi-Required:** Probability that course will be taken > 50%

<b>REQ</b>	<b>Requirements</b>
<b>DES</b>	<b>Design</b>
<b>CST</b>	<b>Construction</b>
<b>TST</b>	<b>Testing</b>
<b>MNT</b>	<b>Maintenance</b>
<b>CNF</b>	<b>Config. Mgmt.</b>
<b>MGT</b>	<b>Management</b>
<b>PRC</b>	<b>Process</b>
<b>TLS</b>	<b>Tools and Methods</b>
<b>QLY</b>	<b>Quality</b>

# SWEBOK Coverage - Required & Semi-Required Courses



# Composite SWEBOK Coverage in Required and Semi-Required Courses



- No consistency in coverage across programs
- Requirements, design, and management are best covered
- Maintenance, configuration management, quality, and tools are least covered

**% of Programs with one or more dedicated courses**

# Innovative Courses

1. Reverse Engineering (*Drexel*)
2. Software Evolution and Re-engineering (*Rochester*)
3. Software Documentation (*Penn State Great Valley*)
4. Software Risk Assessment in DoD (*NPS*)
5. Refactoring (*Mercer*)
6. Structured Document Interchange and Processing (*DePaul*)
7. Avoiding Software Project Failures (*Carnegie Mellon – West*)
8. Mathematical Foundations of Software Engineering (*Monmouth*)
9. Global Software Development (*Carnegie Mellon*)
10. Management of Outsourced Development (*Carnegie Mellon – West*)
11. Professional, Ethical and Legal Issues for Software Engineers (*Cal. State Univ. – Fullerton*)
12. Managing Software Professionals (*Carnegie Mellon – West*)
13. Lean & Agile Software Processes (*Mercer*)
14. Artificial Intelligence (*Michigan - Dearborn*)
15. Software Engineering Economics (*GMU, USC*)
16. Computer Game Design & Implementation (*Michigan - Dearborn*)
17. Service Oriented Architecture (*Dublin City*)