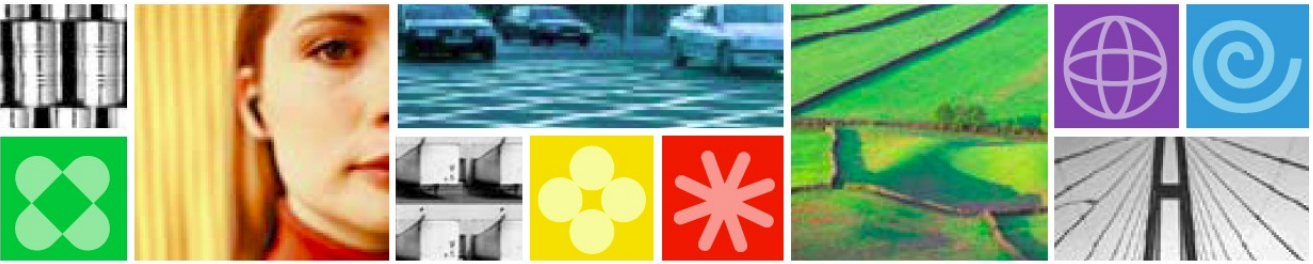
IBM Academic Initiative
System z

EzTeach – Enterprise Computing Education the Easy Way



ibm.com/university/systemz

Kathy Pfeiffer (pfeiff@us.ibm.com) & Angelo F. Corridori (angelo.corridori@marist.edu)

Agenda

- IBM Academic Initiative Curriculum Analysis & Recommendations
- Course Material Objectives and Definition
- Focus Groups – What we Learned
- Course Materials Developed
  - Operating System Functions module
  - Enterprise Computing Insights
- Next Steps

Academic Initiative – Content observations and integration

- **Adoption of content depends on size and type of school**
- **New course – difficult**
  - Full semester
  - Levels of approval
  - Lengthy process
  - Opportunities - special topics / electives
- **Integration of topics – Easier**
  - Module oriented
  - More flexibility
  - Greater implementation
  - Customized to fit program's learning objectives / various degree programs
- **Most educators prefer**
  - Vendor neutral content
  - Hands on component
  - Adherence to accreditation standards and guidelines, as well as state and school policies

Four key questions educators consider when developing / modifying content

- What **objectives** do I hope to accomplish?
- What **topics or content** will I have to cover?
- What **teaching methods** or strategies should I use to direct learning and achieve the objectives?
- How do I **evaluate instruction** to determine whether I have successfully achieved the objective?

Professional and computing societies

Societies have taken a leading role in providing support for higher education in various ways, including the formulation of curriculum guidelines

- The Association for Computing Machinery (ACM)
- The Association for Information Systems (AIS)
  - Most academic members of AIS are affiliated with Schools/Colleges of Business or Management.
- The Association for Information Technology Professionals (AITP)
- The Computer Society of the Institute for Electrical and Electronic Engineers (IEEE-CS)

Accreditation

- ABET- Accreditation Board for Engineering and Technology
- CSAB - Computing Sciences Accreditation Board

Curriculum guidelines and standards

- Today, societies cooperative in creating standards
- Overview Document - Computing Curricula Series (CC2005)
  - Single report, provides undergraduate curriculum guidelines for five defined sub-disciplines of computing

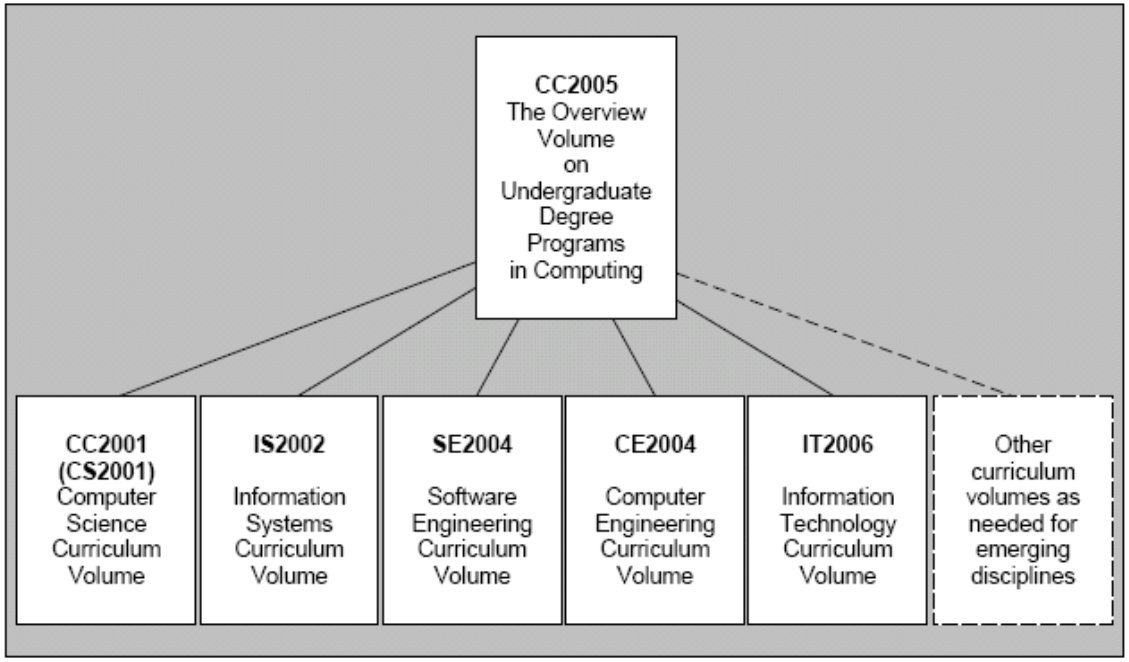
  - Computer engineering (CE)

  - Computer science (CS)

  - Information systems (IS)

  - Information technology (IT)

  - Software engineering (SE)



Degree program characteristics

- Task force identified 40 topics (knowledge areas) that span 5 computing degree programs
- Assigned min / max value for each computing degree program (CE, CS, IS, IT SE)
- Example:
  Knowledge Area: Programming Fundamentals
  Degree program: CS
  Min value: 4, Max Value: 5
- Analysis of min/max values (to prioritize module development)
  - Average comparative weight of 3 or higher
  - Topics that spanned 3 or more degree programs

Knowledge Area analysis

Knowledge Area	CE	CS	IS	IT	SE
Programming Fundamentals	**X**	**X**	**X**	**X**	**X**
Algorithms and Complexity	**X**	**X**			**X**
Computer Architecture	**X**	**X**			**X**
Operating Systems Principles & Design / Configuration and Use	**X (PD)**	**X (PD, CU)**		**X (CU)**	**X (PD, CU)**
Networking Principles and Design / Use and Configuration		**X (PD)**	**X (UC)**	**X (PD, UC)**	**X (PD)**
Human-Computer Interaction	**X**	**X**	**X**	**X**	**X**
Information Management (DB) Theory / Practice		**X (T)**	**X (P)**	**X (P)**	**X (T)**
Legal / Professional / Ethics / Society	**X**	**X**	**X**	**X**	**X**
Analysis of Technical Requirements	**X**	**X**	**X**	**X**	**X**
Software Design	**X**	**X**			**X**
Distributed Systems	**X**		**X**		**X**

Operating Systems comparison

CE OS *Principles and Design* Core hours - 20 Source - CE2004	**CS** OS *Principles and Design* Core hours - 18 Source - CS2008	**IT** *OS Configuration & Use* PT – Core Hours – 10 SA – Core hours – 4 Source – IT2008	**SE** *OS Principles and Design* *OS Configuration & Use* Source – SE2004
History and overview [1] Design principles [5] Concurrency [6] Scheduling and dispatch [3] Memory management [5]	Overview of Operating Systems [2] OS Principles [2] Concurrency, Scheduling and Dispatch [6] Memory Management [3] Security & Protection [3] Real Time and Embedded Systems [2]	**Platform Technologies** Operating Systems [10] *Topics:* Overview Operating system principles Concurrency Scheduling and dispatch Memory management Device management Security and protection File systems Real-time and embedded systems Fault tolerance Scripting Virtualization **System Administration and Maintenance** Operating Systems [4] *Topics:* Installation Configuration Maintenance (service packs, patches) Server services (print, file, DHCP, DNS, FTP, HTTP, mail, SNMP, telnet) Client services Support	**Full semester courses:** Operating Systems and Networking

Recommendations

- When possible, develop / prioritize content that aligns with academic organization and society curriculum recommendations
  - Greater acceptance
- When developing / modifying content priority should be given to:
  - Core requirements (versus electives)
  - Topics that span multiple computing degrees (greater implementation)
  - Topics that highlight the value and benefit of enterprise systems
- Content should be topic related (versus product related)
- Content length should be small modules versus semester length (greater integration)
- Content should contain a hands-on component (lab) or real scenario (case study)

Course Material Objectives

1. are consistent with the ACM, IEEE, AIS, etc. recommendations on what should be included in various technical curricula that include computing topics (e.g. Computer Engineering, Computer Science, IT, Systems Engineering, etc.)
2. highlight the capabilities and strengths of Enterprise systems
3. are ***easy*** for faculty to adopt and include in existing classes

Course Materials – What to Develop?

- A module is:
  - Power Point charts
  - Audio for charts
  - Written notes for charts
  - Book like "chapter" or text that covers the same material as the presentation
  - Quiz questions
- -
  - Required reading (secondary)
  - Optional reading (secondary)
  - Lab (future objective)
- About a week's worth of material for a class (1 hour (preferred) to 2 hours (at most) of presentation)
- High level (avoid the technical details)
- Emphasize the unique capability of System z & z/OS

Focus Groups

- Marist Student Focus Group
  - How to present Enterprise Computing concepts?
  - What materials are most effective?
- Recent Hire Focus Group
  - Based on your working experience, what should have been covered in your undergraduate courses, but wasn't?
- ------ Module Development -------
- Marist Faculty Focus Group
  - Can you use these materials?
- Marist student Focus Group
  - What do you think of the materials?

Marist Student Focus Group – What we learned

- Reviewed Project Background and Objectives
- Suggested ways to present Enterprise Computing content
  - Examples of technology pioneered by mainframes (Virtual Storage, multi-processing, etc.)
  - Novel/Atypical uses of mainframes (multi-user gaming, real time crime analysis)
  - Technical Case Studies (back of text book)
- What content/format is most effective?
- Feedback
  - Case studies in texts referred to occasionally but not used in class
  - Classes cover primarily theory/concepts (vs. implementation)
  - Implementation examples would be helpful in understanding concepts
  - Few Enterprise Computing concepts covered (major omissions – e.g. security)
  - Linux used as example in OS class text book
  - Power Point charts (from text) mostly used in class; text itself was a reference, not heavily used
  - Get some materials into the existing text books
  - Show an end to end example of a real request

Recent Hire Focus Group – What we learned (What do you wish had been covered?)

1. I/O configurations (complex)
2. High level view of how things "fit together"
3. Recovery
4. *Serviceability
5. *Debugging / working with someone else's code
6. *Importance of commenting code (related to #5)
7. *Different models of computing (Distributed, yes, Centralized, no)
8. *GRID & Cloud
9. *Transaction processing; two phase commit, in flight, in doubt transactions
10. *Front end web development

Recent Hire Focus Group – What we learned

11. Security
12. *Testing U/T, F/T, System Test, etc.
13. *Write once but service multiple types of users/clients (e.g. phone, laptop, iPad, etc.)
14. *Code reuse.
15. Release to release toleration designing for future extension/enhancement.
16. *Use existing libraries; don't write everything from scratch
17. *Database programming
18. *Data integrity
19. *Languages other than JAVA
20. *Virtualization

Recent Hire Focus Group – What we learned

21. *Build, Integration, Dependencies in large software projects
22. *Project Management and working within and with schedules of others

* - Topics NOT covered in ACM CS/ITS recommendations

Operating System Functions Module Content

- Topic Chosen: High Level View of How Things Fit Together (in an operating system)
- Agenda for Module developed
  - Introduction
  - Operating System Design Points
    - Personal Computer Laptop Design Points
    - Enterprise Server Design Points
  - Common Operating System Function
    - E.g. Process Mgt., Multi-tasking, Memory Mgt., I/O, etc.
  - Unique Operating System Functions
    - Personal Computer Laptop Unique Functions
      - E.g. Plug & Play Mgr., Install Wizard, GUI, Hardware Abstraction Layer, etc.
    - Enterprise Server Unique Functions
      - E.g. WLM, Clustering, JES, TSO, USS, RTM, etc.
  - Review

Marist Focus Groups – What we learned

- Faculty / Students reviewed an early version of the OS Functions presentation (only)
- Feedback
  - Faculty
    - Use entire module “as is” in ITS 130 class
      - Reduce 45 min presentation to 20 min?
    - Use portions of module in Computer Architecture class
    - Possible use of some charts in project based operating system class
  - Students
    - Presentation flowed well
    - Liked use of analogy
    - Like Adobe Presenter for delivery of presentation
    - Liked OS structure diagrams at end of presentation to summarize concepts
  - Developer (me)
    - Difficult to develop the material since you don’t know the audience & their background

Marist Faculty Usage Spring ‘12 – What we learned

- Time constraints prevented even motivated faculty from using the presentation or parts of the presentation in their classes
  - 5 faculty approached
  - Help offered (present, help prep, etc.)
- One faculty member used one chart in one class

Operating System Functions Module

- Instructor Materials
  - Instructor Overview of Materials provided
  - Suggested uses for module and materials
  - Quiz Questions and Answers (25 questions)
  - Discussion Questions (11 questions)
  - Reference Materials (based on commonly used texts and readily available IBM materials)
- Student Materials:
  - Student Topic Instructions
    - sample instructions - may be modified by the instructor
  - PowerPoint presentation (segmented) with audio and slide notes
  - Text Book Chapter (20 pages)
  - Quiz Study Guide
  - Quiz (25 questions)
- Packaging
  - Adobe Portfolio – single file
  - Free Adobe reader (V9+) needed to access and use materials

Hmmm……

- Is a one hour module too long?
- Are ***more*** instructor aids (presentation, audio, notes, chapter, quiz, references, etc.) ***less*** usable?
- What materials can be developed that eliminate or drastically reduce the time needed for the instructor to include Enterprise Computing concepts?

Enterprise Computing Insights (ECI)

- What is it?
  - A Short (2-3 page) article
  - Describes one Enterprise Computing concept
  - Easy to read & understand (10-15 min)
  - Relates to familiar concepts (e.g. from a PC or everyday experiences (e.g. using an ATM))
  - Assumes the reader knows nothing about the concept
  - Intended to make Enterprise Computing approachable (and interesting)
  - Cross referenced to each other
  - Relates to ACM, IEEE, etc. terms & concepts where applicable
- What it is not
  - Complete, Thorough treatment of the concept

Enterprise Computing Insights (cont.)

- Concepts (so far)
  - What is Enterprise Computing?
  - What is Centralized Computing?
  - What is Transaction Processing?
  - What is Batch Processing?
  - What is an Enterprise Server?
  - What is a Data Set?
- Minimal Instructor Effort, Low risk
  - Select a topic; read the topic (15 min); discuss the topic

Why I Like ECI Approach

- **Simple**
  - No technology (sheet of paper)
  - One deliverable, not many
  - Easy to decide whether or not to use each one
  - Little thought needed to decide how to use it (suggestions provided)
- **Little or no work for the instructor to get started**
- **Flexible**
  - Can be used in any order
  - Can be used in multiple ways (see suggestions for use)
  - Can be used in any relevant class
  - Can be used at any undergraduate level (Freshman-Sr.) & possibly High Schools
- **Relates to appropriate rigorous material requirements from ACM, IEEE, etc.**
  - e.g. refers to ACID for transaction processing

Why I Like ECI Approach (cont.)

- **Provides more opportunity for "active learning" for both the instructor and student than standard Power Point presentation + reading**
  - discussions
  - presentations
  - writing

Next Steps

- **Faculty Feedback on two approaches**
  - ECC work session to examine
    - OS Functions module
    - ECIs
  - IBM T3 work session to examine
    - OS Functions module
    - ECIs
- **Work with IBM AI to set direction for future development based on faculty feedback**

Questions?

Backup

Knowledge Area Definitions

- **Algorithms and Complexity** – Computational solutions (algorithms) to problems; time and space complexity with respect to the relationship between the run time and input and the relationship between memory usage and input as the size of the input grows.
- **Computer Architecture and Organization** – Form, function, and internal organization of the integrated components of digital computers (including processors, registers, memory, and input/output devices) and their associated assembly language instructions sets.
- **Distributed Systems** – Theory and application of multiple, independent, and cooperating computer systems.
- **Human-Computer Interaction** – An organizational practice and academic field of study that focuses on the processes, methods, and tools that are used for designing and implementing the interaction between information technology solutions and their users.
- **Information Systems Development** – The human activities -- including requirements analysis, logical and physical design, and system implementation -- that together lead to the creation of new information systems solutions.

- **Integrative Programming** –Uses the fundamentals of programming to focus on bringing together disparate hardware and software systems, building a system with them that smoothly accomplishes more than the separate systems can accomplish.
- **Legal / Professional / Ethics / Society** – The areas of practice and study within the computing disciplines that help computing professionals make ethically informed decisions that are within the boundaries of relevant legal systems and professional codes of conduct.
- **Net Centric: Principles and Design** – Includes a range of topics including computer communication network concepts and protocols, multimedia systems, Web standards and technologies, network security, wireless and mobile computing, and distributed systems.
- **Net Centric: Use and Configuration** – The organizational activities associated with the selection, procurement, implementation, configuration, and management of networking technologies.
- **Operating Systems Principles & Design** – Underlying principles and design for the system software that manages all hardware resources (including the processor, memory, external storage, and input/output devices) and provides the interface between application software and the bare machine.

- **Operating Systems Configuration & Use** – Installation, configuration, and management of the operating system on one or more computers.
- **Programming Fundamentals** - Fundamental concepts of procedural programming (including data types, control structures, functions, arrays, files, and the mechanics of running, testing, and debugging) and object-oriented programming (including objects, classes, inheritance, and polymorphism).
- **Software Design** - An activity that translates the requirements model into a more detailed model that represents a software solution which typically includes architectural design specifications and detailed design specifications. [Alternatively, in software engineering, the process of defining the software architecture (structure), components, modules, interfaces, test approach, and data for a software system to satisfy specified requirements. [ANSI/IEEE Standard 729-1983]]
- **Software Modeling and Analysis** – An activity that attempts to model customer requirements and constraints with the objective of understanding what the customer actually needs and thus defining the actual problem to be solved with software.
- **Analysis of Technical Requirements** - The process through which a computing development project determines the computing and communications hardware and software based on the goals of the individual user(s) or the user organization(s).
- **Theory of Programming Languages** – Principles and design of programming languages including grammars (syntax), semantics, type systems, and various language models (e.g., declarative, functional, procedural, and object-oriented).