Improving Software Architecture Competence

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Paul Clements
29 March 2007

Software Engineering Institute | Carnegie Mellon

Agenda

Part 1. Introduction

Part 2. Duties, skills, and knowledge of software architects

Part 3. A human performance model for architecture competence

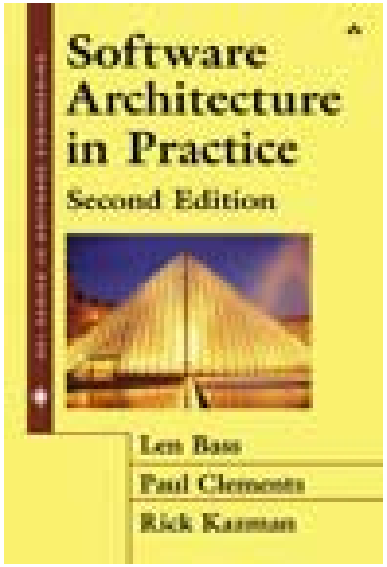
Part 4. Conclusions

The Ascendance of Software Architecture

Over the past 10 years, software architecture has emerged as the prominent paradigm in large-system development.

There are:

- worldwide conferences devoted to it
- books devoted to it
- defined “architect” roles in organizations
- courses and training for it
- professional organizations supporting it
- journals dedicated to it



Software architecture

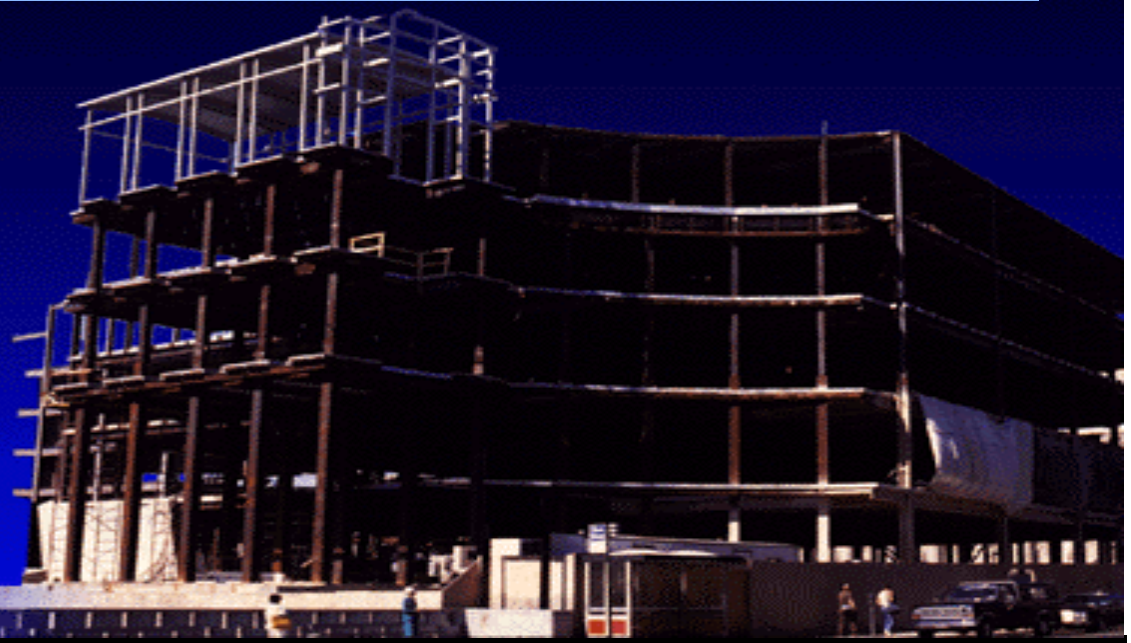
The rise of software architecture has resulted from two trends:

- Recognition of the importance of quality attributes
- The development of very large and very complex systems

What Is Software Architecture?

Software architecture is the structure or structures of the system, which comprise software elements, the externally visible properties of these elements, and the relationships among them.

-- Bass, L.; Clements, P.; & Kazman, R. *Software Architecture in Practice, Second Edition.* Boston, MA: Addison-Wesley, 2003.



From the technical to the personal

Most of the work in architecture to date has been technical

- Design and creation
- Evaluation and analysis of architectures
- Styles and patterns
- Architecture-based testing
- Architectural reuse and software product lines
- Architectures for particular domains
- Architectural re-engineering and recovery

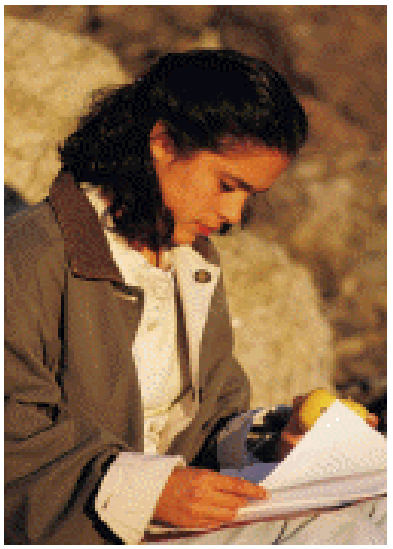
This accounts for an impressive body of work in a short time.

As part of the continuing maturation of architecture, to make it an engineering discipline, it is time to turn our attention to this fact:

Architectures are created by *people* working within *organizations*.

How can we help them do what they do better?

New project: Improving Software Architecture Competence



Architectures are created by *architects*.

- How can we help them do their best work?
- What does it mean for an architect to be competent?
- How can an architect improve his/her competence?

Architects work in *organizations*.

- How can we help an organization help their architects do their best work?
- What does it mean for an organization that produces architectures to be competent?
- How can an organization improve its competence in architecture?

Competence

Competent: Capable of performing an allotted or required function.

- Source: The American Heritage® Stedman's Medical Dictionary, Published by Houghton Mifflin Company, 2002.

Proposal:

A competent architect (architecting organization) is one that carries out his/her (its) architecture-related duties competently.

Duties may be hindered by an incompetent organization, but this gives us a way to evaluate architects and organizations in the here-and-now.

Agenda

Part 1. Introduction

Part 2. Duties, skills, and knowledge of software architects

Part 3. A human performance model for architecture competence

Part 4. Conclusions

What do architects do?

Philippe Kruchten writes that he requires architects working for him to spend 50% of their time on the architecture.

What do they do with the *other* 50%?

To understand how to help architects do what they do, we need to understand what they do.

- What are their duties?
- What skills and knowledge made them "capable of performing their allotted or required function?"

How can we find this out?

We can survey the "community"

Three broad sources of information (with count so far)

- "Broadcast" sources: Information written by self-styled experts for mass anonymous consumptions
  - Web sites: e.g., Bredemeyer, SEI, HP, IBM (16*)
  - Blogs and essays (16*)
  - "Duties" list on SEI web site
  - Books on software architecture (25 top-sellers)
- Education and training sources:
  - University courses in software architecture (29*)
  - Industrial/non-university public courses (22*)
  - Certificate and certification programs in architecture; e.g., SEI, Open Group, Microsoft (7*)
- "Architecture for a living" sources
  - Position descriptions for software architects (60)
  - Résumés of software architects (12)
  - Questionnaires from practicing architects (30+, not yet processed)

* Exhaustive or near-exhaustive web search

Example: Course description

Software Architecture Workshop	Small teams work on creating a draft architecture using the Visual Architecting Process. We follow the iterative architecting process, and weave concepts and architecting techniques into the lectures between work sessions. Topics: • Software architecture: meta-architecture, architectural patterns, architecture modeling using the Unified Modeling Language (UML); architectural views; component specification, key architectural design principles; • The architecting process: Architectural requirements, system specification, architecture validation; • Organizational process: Sponsorship, leadership, consulting. • Role and responsibilities of the architect: relates the responsibilities and associated skills and attitudes of the architect to the architecting process. • • http://www.bredemeyer.com/architecture_workshop_overview.htm •
Duration	
4 days	
Cost	
$2250	

Example: Position description

Infosys Technologies, Ltd., India, 3 May 2006

Responsibilities :

- You will be required to define architecture and provide highly technical direction to high priority or special projects.
- You will be responsible for evaluating new technological development and evolving business requirements.
- Providing in-depth technical and business knowledge to ensure efficient design, programming and implementation will be a key area of focus.
- You will have to understand performance issues and approach them systematically,
- Conduct technical studies and evaluations of business area requirements and recommend appropriate technological alternatives.
- You will have to conduct independent analysis of business models, logical specifications and/or user requirements to design business solutions along with technical studies and evaluations of business area requirements and recommend appropriate technological alternatives.
- You will assist senior technology and architecture staff within the portfolio in determining the direction of current and future programming/systems initiatives,
- Recommend new technologies and products which enhance operations and functionality of systems

Qualifications : BE / MCA / MSC

Experience / Skills :

6-12 Years overall experience

- Databases - Ab Initio, Cognos, DB2, Essbase, Informatica, MicroStrategy, MS Access, MySQL, Oracle, SQL Server, UDB
- Programming Languages - C, C# (C Sharp), C++, VC++/MFC, Visual Basic (VB) / VB.NET
- Software - Business Processes
- Web Technologies - ASP / ASP.NET, JAVA, Java Server Pages (JSP), XML Technologies
- Web / Middleware Technologies: Java, EJB, J2EE, JSP, Web Services, SOAP, CORBA, Orbix, VisiBroker, XML, J2ME, MQSeries, TIBCO, Websphere, NetDynamics, WebLogic, SunOne, Zope, PlumTree
- IBM AS / 400: RPG/400, COBOL/400, DB/400, COOLPLEX
- Databases: Oracle, Sybase
- Microsoft Technologies:VB, ASP, IIS, MTS, Crystal Reports, VC++, .NET, PL/SQL, SQLServer, BizTalk
- IBM Mainframes: COBOL, JCL, CICS, DB2, IMS, IDMS, Natural, ADABAS, REXX, Assembler, CLIST, QMF, MVS, OS390, OS/2, VSAM, QSAM, Sysplex
- Datawarehousing / Business Intelligence: Teradata, Business Objects, Cognos, Informatica, Crystal Enterprise Suite, Actuate, Ab Initio, DataStage, SAS, MicroStrategy
- Open Systems: C, C++, Perl, Solaris, Linux, PRO*C, Unix, Windows
- Miscellaneous: Remedy, MfgPro, Vignette, Filenet, Documentum, IBM Tivoli, Netegrity, Lotus Notes, MUMPS, Cerner, Facet
- Exposure to formal quality processes and a strong foundation in SDLC concepts are necessary.
- Expertise in German / French / Japanese (JLPT-L2 & above) will be an added advantage.

Survey results to date

To date, we have surveyed over 200 sources.

We have cataloged

- 201 duties
- 85 skills
- 96 knowledge areas

Advantages of survey approach

It cleanly avoids “definition wars” about architects and architectures. We don’t have to stake out a position on what architects do and know. The community tells us.

We do not care what career path a person has taken to become an architect (volunteered, drafted, informal…)

It’s systematic. Given a kind of source, we can go take a meaningful sample of it.

Challenges of survey approach

Bewildering variety of job titles

- IT architect, solution architect, software systems architect, enterprise architect, Java architect, middleware architect, platform architect, enterprise architect, even “code architect.”
- Solution:

Position description for “architect” but clearly using the term only as a prestigious title for a coder

Assigning data to categories

- If you “do” it, it’s a duty
- If you “are” it, it’s a skill
- If you “know” it, it’s knowledge

Handling like-sounding entries

- “Document the architecture” / “Write down the architecture”
- Solution: Treat every contribution as unique, and use an affinity diagram to clump similar things together

Affinity diagrams

Originally developed by Jiro Kawakita, an anthropologist, to discover meaningful groups of ideas from a raw list. Kawakita's idea is to examine the list and let groupings emerge naturally, using the right side of the brain, rather than following a pre-ordained categorization.

Steps

- Assemble the team.
- Write individual statements on note cards.
- Group the statements.
- Name each group.
- Cluster the groups.

We ran three affinity diagram exercises, taking 8 hours over 3 days.

Example duties

<snip>

Duties related to documentation

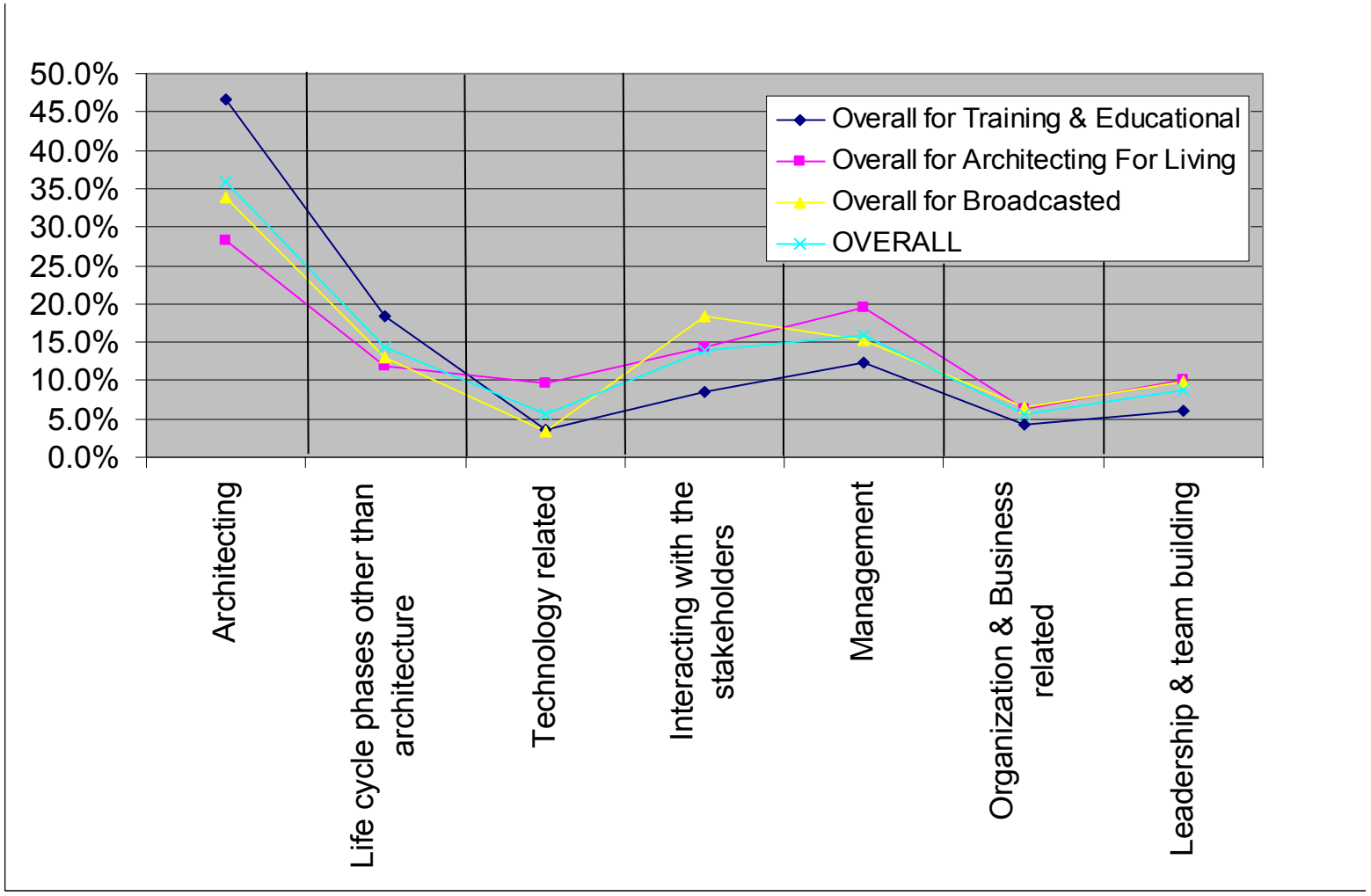
- Thoroughly understand and document the areas (domains) for v system will be built
- Prepare architectural documents and presentations
- Document software interfaces
- Produce a comprehensive documentation package for architecture useful to stakeholders
- Keeping reader's point of view in mind while documentation
- Creating, standardizing and using architectural descriptions
- Use a certain documentation standard
- Document variability and dynamism
- Create conceptual architectural view

<snip>

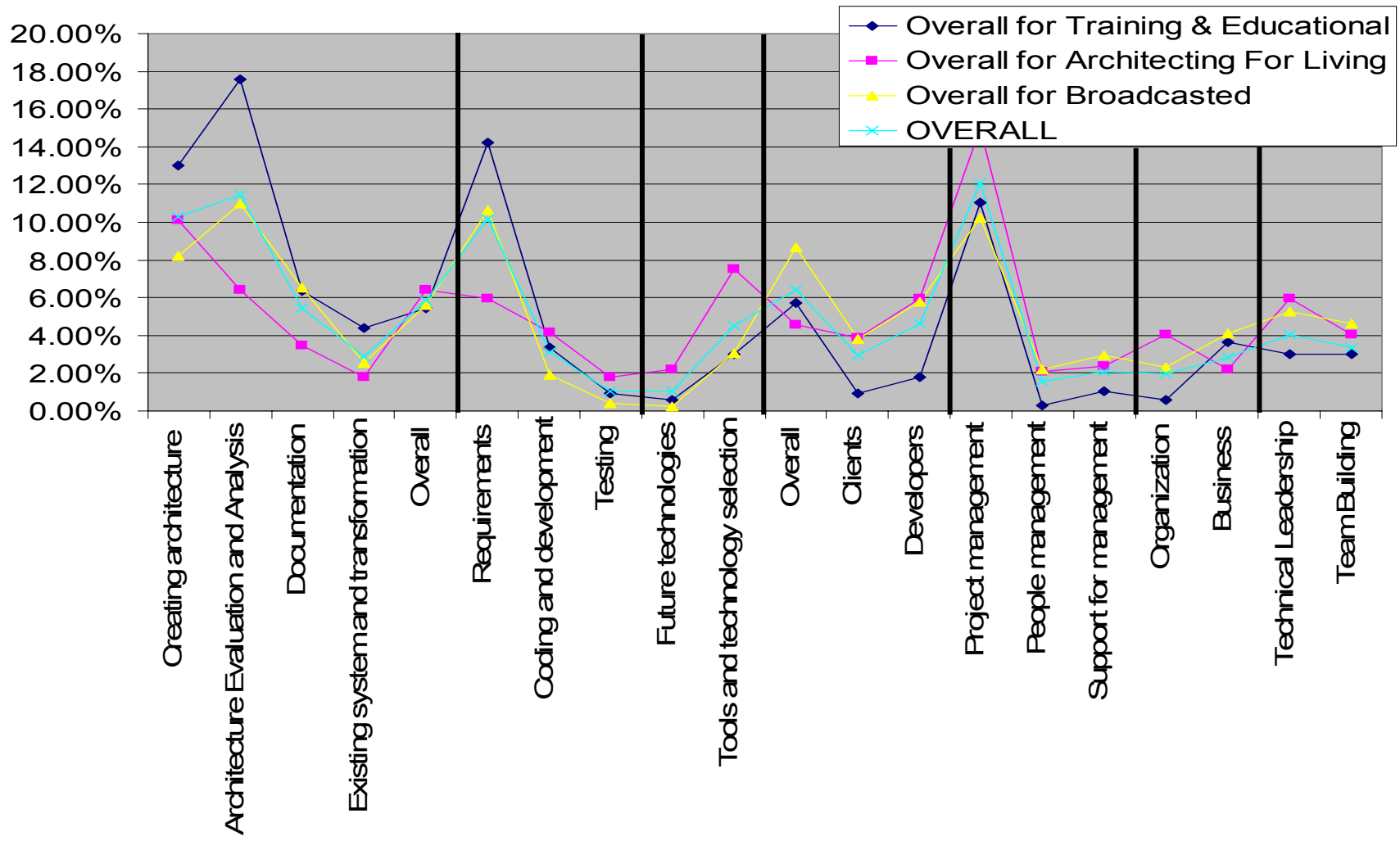
Architectural duty clusters

Architecting	• Overall • Creating the architecture • Architecture evaluation and analysis • Documentation • Existing system and transformation
Life cycle phases other than architecture	• Requirements • Testing • Coding and development
Technology related	• Future technologies • Tools and technology selection
Interacting with stakeholders	• Overall • Clients • Developers
Management	• Project management • People management • Support for project management
Organization and business related	• Organization • Business
Leadership and team building	• Technical Leadership • Team Building

Architecture Duties (categories)



Architecture Duties (sub-categories)



Observations about duties

1. None of the subcategories seems to be considered of paramount importance by any of the sources groups. Everyone agrees: Architects do a lot of different things!
2. Project management, requirements, and architecture evaluation and analysis are the most often mentioned duties overall.
3. The overall curve almost overlaps with the curve corresponding to the broadcast sources.
4. The training and education sources produce a noticeably different profile than the 'architecting for a living' sources. See especially architecture evaluation and architecting.
5. There seems to be a consensus about the "Overall" sub-category of architecting, as well as the categories of organization and business related duties, and leadership and team building.
6. Treatment of stakeholders varies noticeably. "Architecting for a living" sources barely mention interacting with clients. The broadcast sources mention interacting with stakeholders almost three times as much as the "architecting for a living" sources.

Example skills

<snip>

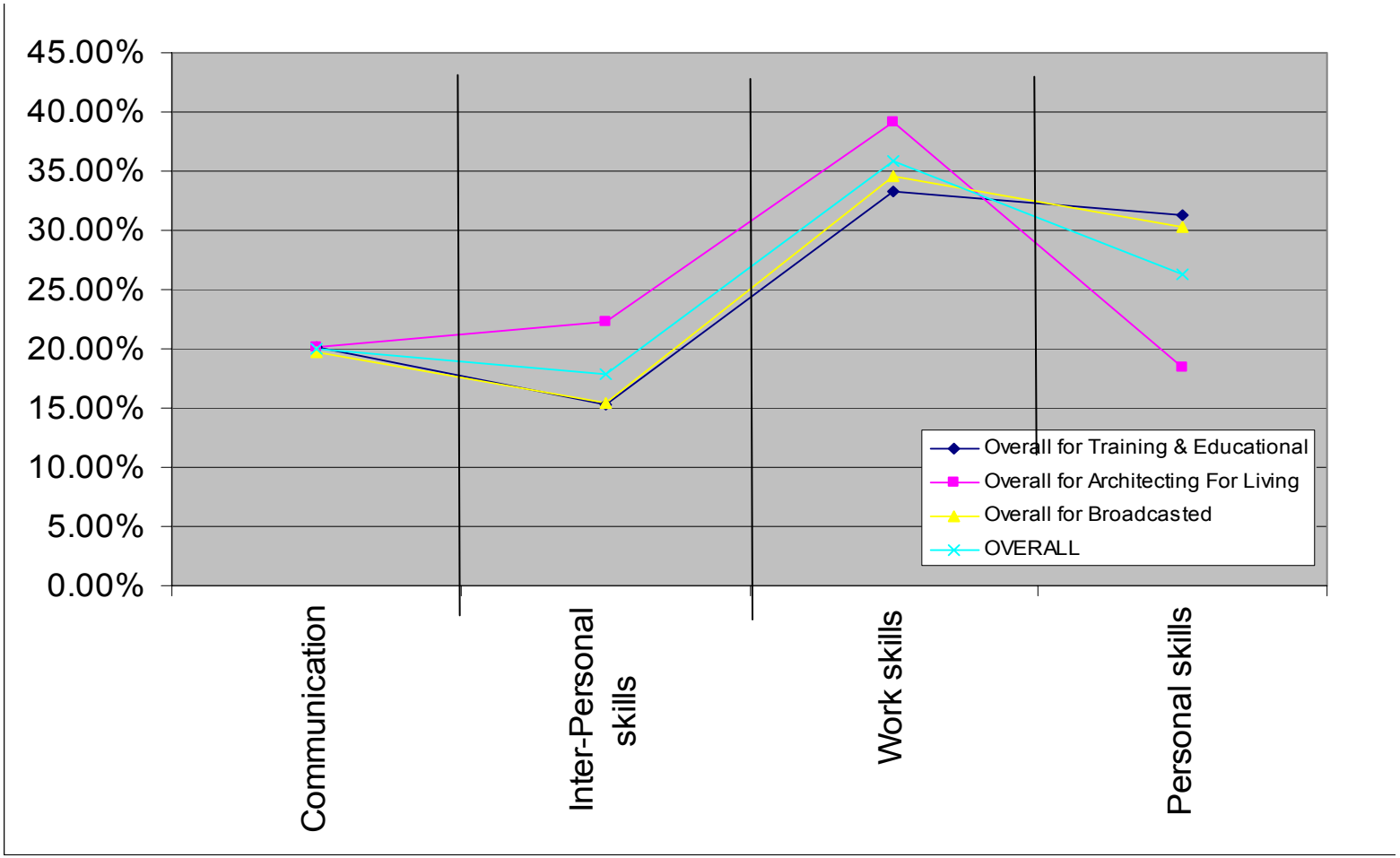
- Inspire creative collaboration
- Interpersonal skills
- Interviewing
- Investigative
- Leadership
- Learning
- Listening skills
- Maintains constructive working relationships
- Mentoring
- Negotiation skills
- Observation power
- Open minded
- Oral and written communication skills
- Organizational and workflow skills
- Patient
- Planning skills
- Political sagacity

<snip>

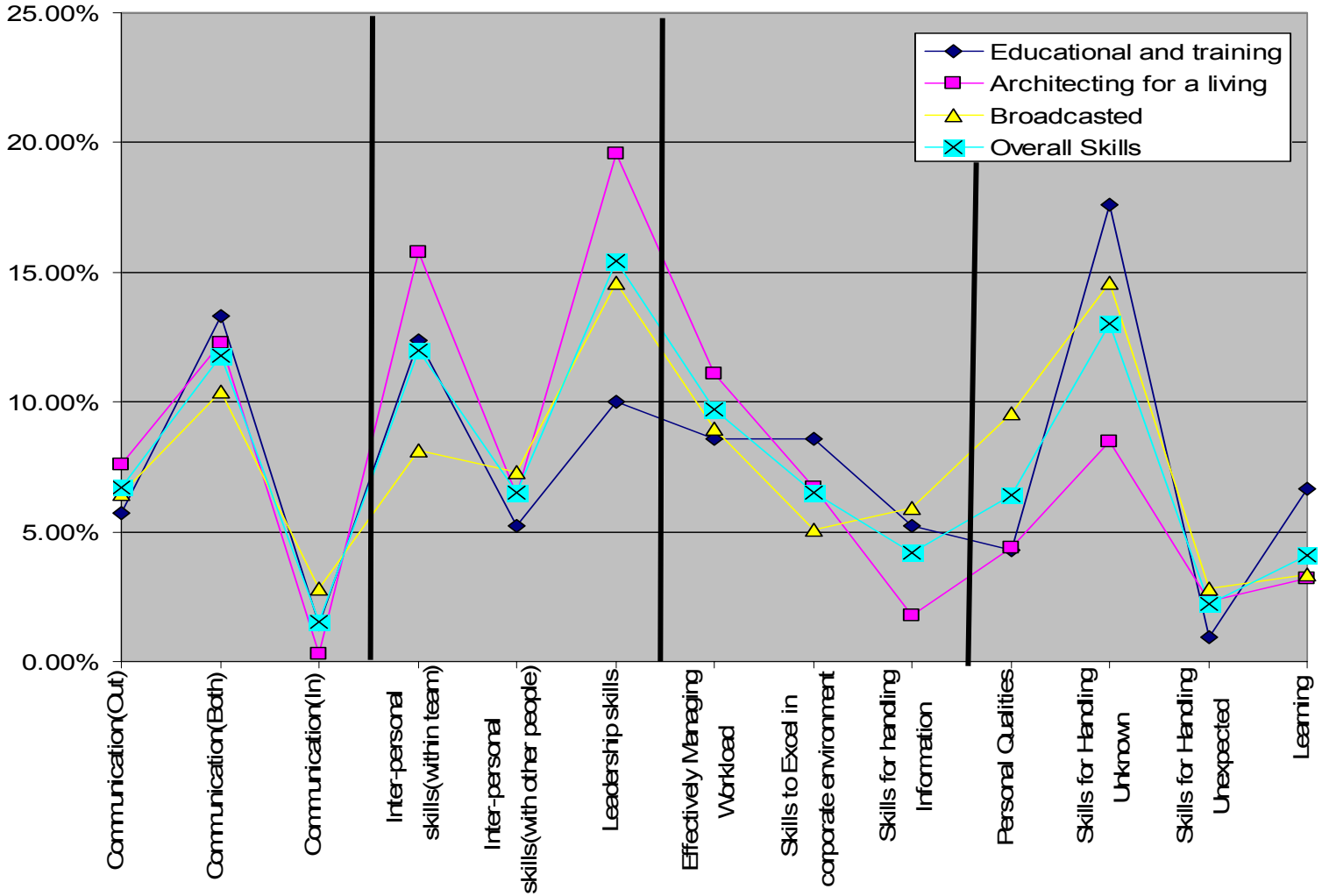
Architectural skill clusters

Communication skills	Out
	Both (i.e., two-way)
	In
Inter-personal skills	Within team
	With other people
	Leadership skills
Work skills	Effectively managing high workload
	Skills to excel in a corporate environment
	Skills for handling large amounts of information
Personal skills	Personal qualities
	Skills for handling unknown
	Skills for handling unexpected
	Learning

Architecture Skills (categories)



Architecture Skills (sub-categories)



Observations about skills

1. All the curves agree on communication. "Communication – in" is barely mentioned.
2. 2. "Architecting for a living" sources mention personal skills the least when compared with other skills or with other sources. At the same time, they place the most emphasis on work skills.
3. The overall curve follows the broadcast curve very closely. However, training and education sources and "architecting for a living" sources are poles apart. Gap between teachers and practitioners?
4. Overall, leadership skills and skills for handling the unknown are the two most often cited.
5. "Architecting for a living" sources give maximum weight to leadership skills, and minimum weight to skills of handling the unknown. In contrast, the training and education sources completely differ and assign maximum weight to skills for handling the unknown and minimum weight to leadership skills, when compared with other sources.
6. Skills for handling the unknown and skills for handling the unexpected are two different but related skill sets. But training and education sources assign the former a score of about 18% while the latter less than 1%.
7. 7. All the sources almost converge for the following skills: skills for handling unexpected, effectively managing high workload, inter-personal skills, and communication skills.

Example knowledge

<snip>

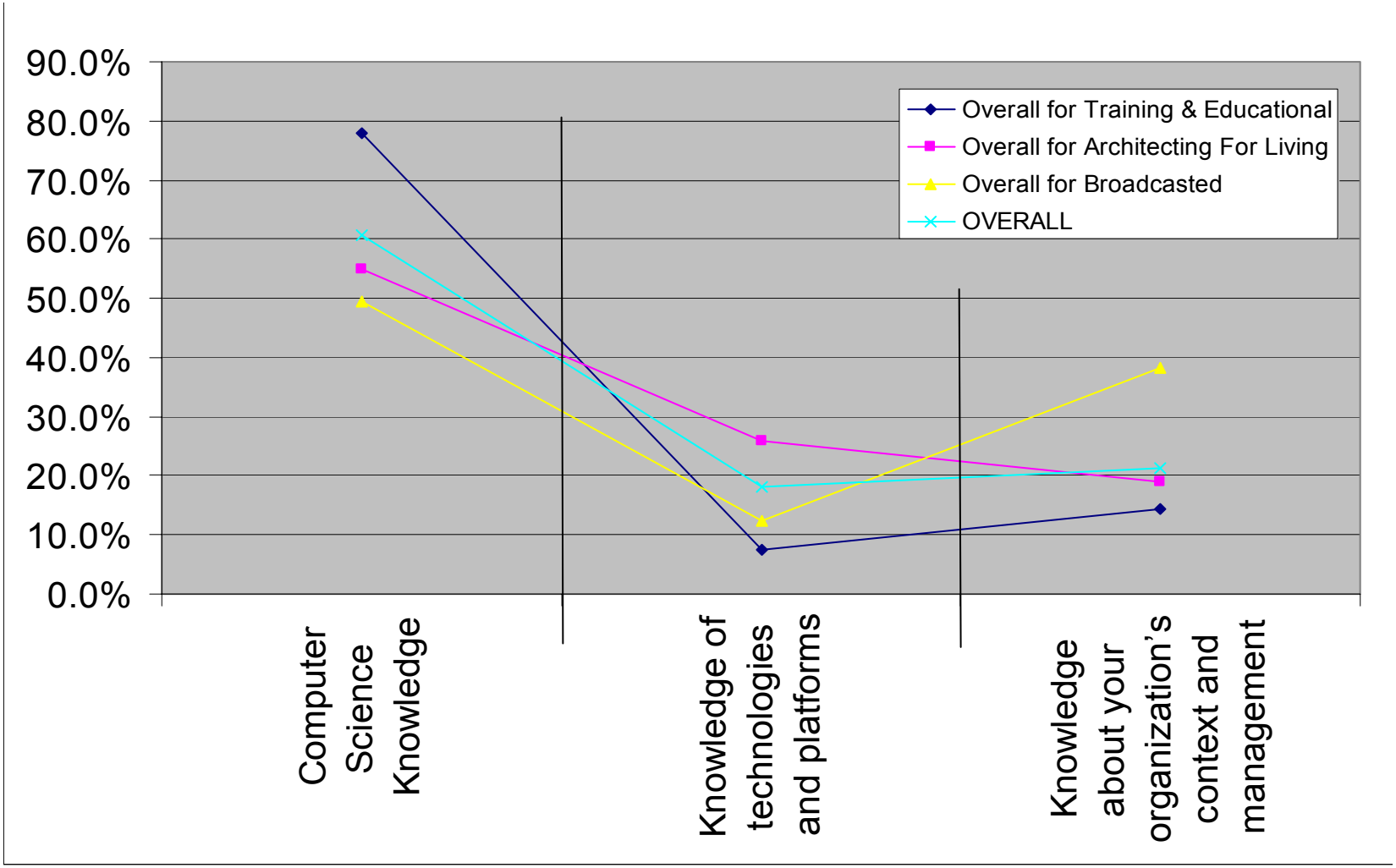
- Software Architecture concepts
- UML diagrams and UML analysis modeling
- Basic knowledge of Software Engineering
- Specialized knowledge of software engineering
- Knowledge about IT industry future directions
- Understanding of web-based applications
- Experience with Web Services Technologies
- Business re-engineering principles and processes
- Knowledge of industry's best practices
- Experience in testing
- Knowledge of testing/debugging tools
- Experience with Real-time systems, Video systems
- Security domain Experience

<snip>

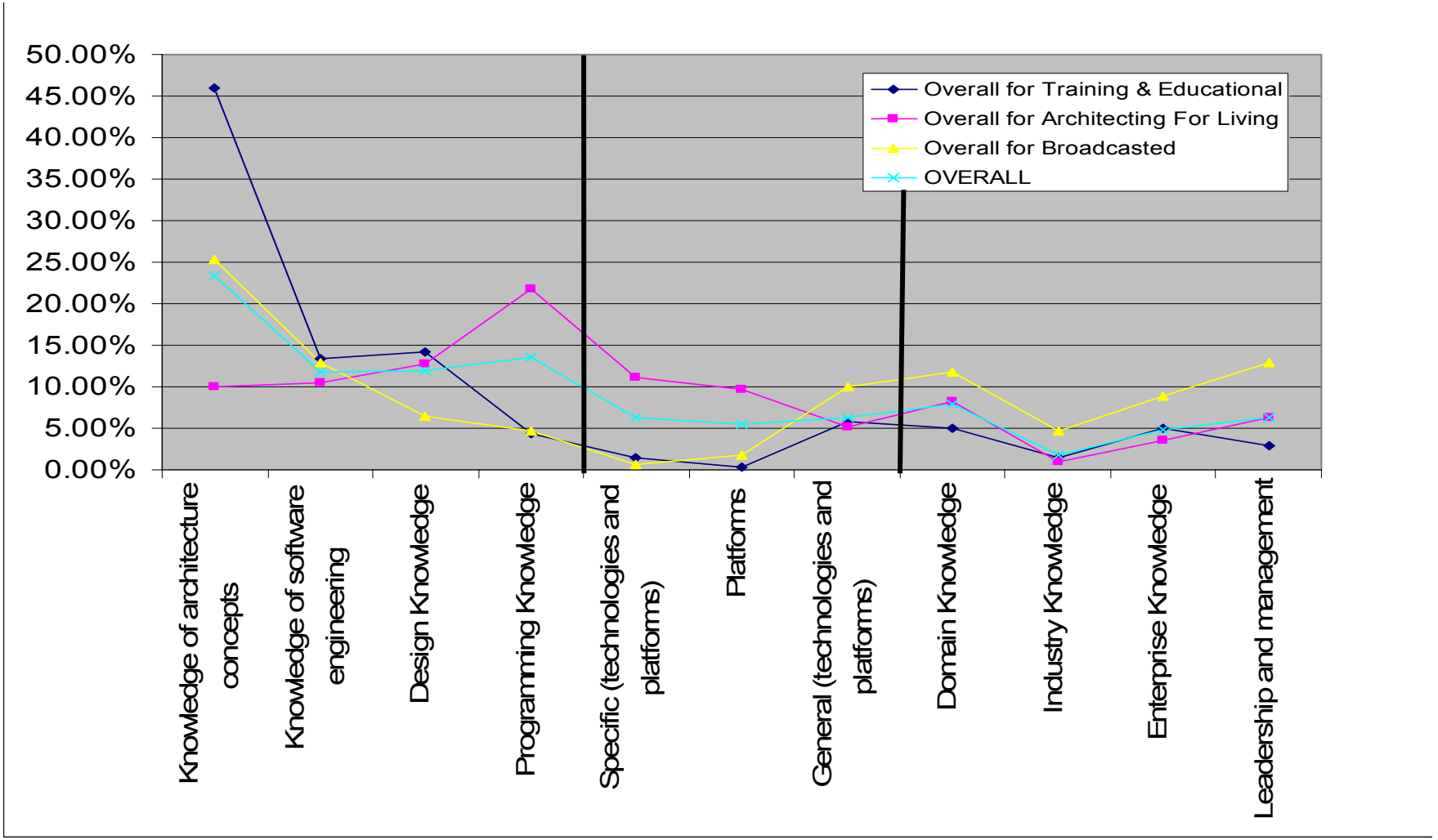
Architectural knowledge clusters

Computer science knowledge	Knowledge of architecture concepts Knowledge of software engineering Design knowledge Programming knowledge
Knowledge of technologies and platforms	Specific Platforms General
Knowledge about organizational context and management	Domain Industry Enterprise knowledge Leadership and management

Architecture Knowledge (categories)



Architecture Knowledge (sub-categories)



Observations about knowledge

1. All the sources consider computer science knowledge to be the most worth mentioning. Training and education sources, as expected, weigh in the heaviest at 80%, while overall it is 60%. The major contributor to this category is knowledge of architecture concepts. It seems this is what architects learn in universities and public courses; other knowledge is generally picked up elsewhere.
2. Training and education sources and “architecting for a living” sources differ by 4x when it comes to knowledge of architecture concepts.
3. “Architecting for a living” sources consider programming knowledge to be the most important knowledge, five times as important when compared with other sources.
4. Sources agree on knowledge of software engineering, industry knowledge, and general knowledge of technologies and platforms.
5. Both training and education sources and “architecting for a living” sources give industry knowledge only around 1% score.
6. Leadership and management is poorly represented by all the sources as knowledge area, whereas it turned out to be one of the most important skills. This seems to validate the idea that leadership and management is more of a skill, not a knowledge area.

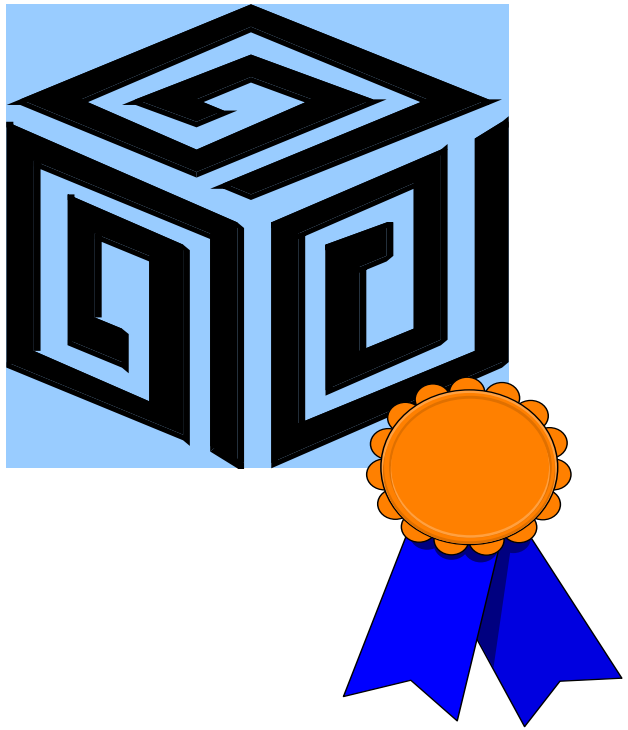
Duties/Skills/Knowledge and Competence

This work lets us propose a "duties/skills/knowledge" model of competence.

Knowledge and *skills* support carrying out the *duties*.

Competence is

- Carrying out the duties
- Having the skills
- Knowing the knowledge



Duties/Skills/Knowledge

Advantages

- It applies equally well to individuals, teams, and organizations.
- It straightforwardly suggests an assessment instrument.
- It straightforwardly suggests an improvement strategy
  - Improve your duties
  - Improve your skills
  - Improve your knowledge

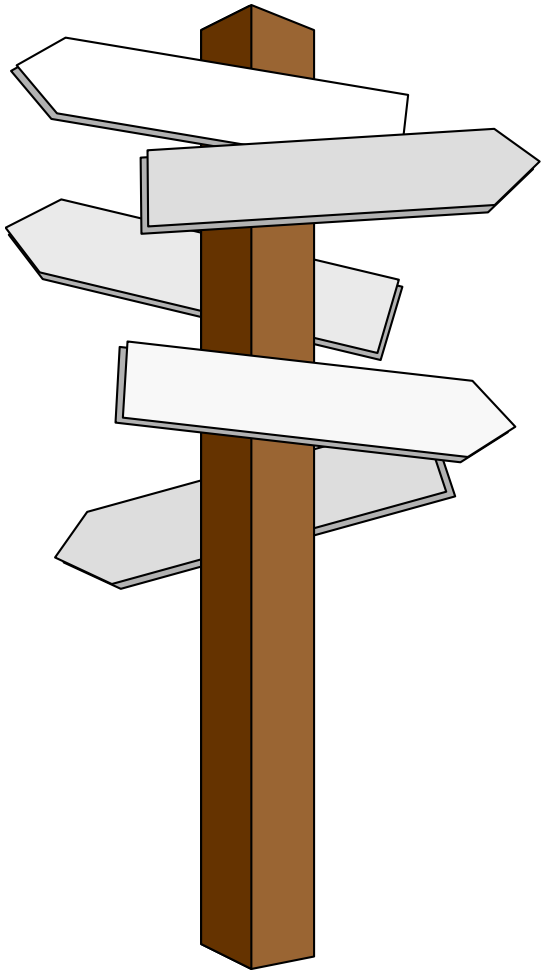
Future work: Apply model to organizations

Case studies and surveys

- organizational excellence in architecture
- organizational improvement in architecture
- surveys of organizational practices

Investigation of *team* practices

Assessment of past performance to find targeted areas of improvement



What are an *organization's* duties, skills, and knowledge?

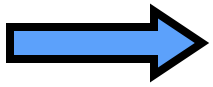
List may include:

- Hire talented architects
- Establish a career track for software architects
- Make the position of architect highly regarded through visibility, reward, and prestige
- Establish a clear statement of duties, responsibilities, and authority for software architects
- Establish a mentoring program for architects
- Start an architecture training and education program
- Track how architects spend their time
- Establish an architect certification program
- Measure architects' performance
- Provide a forum for architects to communicate, and share information and experience
- Put in a place organization-wide development practices centered around architecture
- Establish and empower an architecture review board
- Measure quality of architectures produced
- Initiate software process improvement or software quality improvement practices

Agenda

Part 1. Introduction

Part 2. Duties, skills, and knowledge of software architects



Part 3. A human performance model for architecture competence

Part 4. Conclusions

Gilbert's "Human Competence" work

Thomas Gilbert (1927-1995) is regarded as the "father of human performance" work

- Thomas F. Gilbert, *Human Competence – Engineering Worthy Performance.* HRD Press, Inc., 1996 "Tribute Edition." Book originally published 1978.

Gilbert strongly advocates measuring performance, not knowledge or behavior or motivation or skills or....

- "If I want to know if people are competent, I have to observe how they behave, don't I? My answer to such questions is a firm 'No!'"
- Worth = Value of result / Cost to achieve it. W = V / C
- Egyptian pyramids are "monuments to useless knowledge"
- Arabic alphabet was a much more "worthy" achievement

Measuring Worthy Performance: W = V / C

Performance (or the *worth* of the result) has the following dimensions or "requirements":

Quality

- **Accuracy**: Degree to which accomplishment matches a model, without errors of omission or commission.
- **Class**: Comparative superiority of an accomplishment beyond mere accuracy. Possible measures include market value, judgment points (as for show dogs), physical measures (such as number of mfg. flaws), opinion ratings (Oscars, "MVP")
- **Novelty**: An engine that gets 100mpg is novel. For artistic novelty we probably resort to judgmental points or opinion rating.

Quantity (or Productivity)

- **Rate**: Applies when bulk is time-sensitive; pieces produced per hour; time to completion
- **Timeliness**: Time, not bulk, is key: letter mailed by sundown, Cinderella home by midnight
- **Volume**: Bulk is important, but not time-sensitive. "How many fish did you catch?"

Cost

- **Labor** (behavior repositories): Includes direct overhead, benefits, wages, insurance, taxes
- **Material** (environmental support): Includes supplies, tools, space, energy
- **Management**: Supervision, its supports, public taxes, internal allocations of admin costs.

Measurement and the Performance Audit

First measurements (Institution's accomplishments)

Second measurements (Job accomplishments)

Third measurements (Task accomplishments)

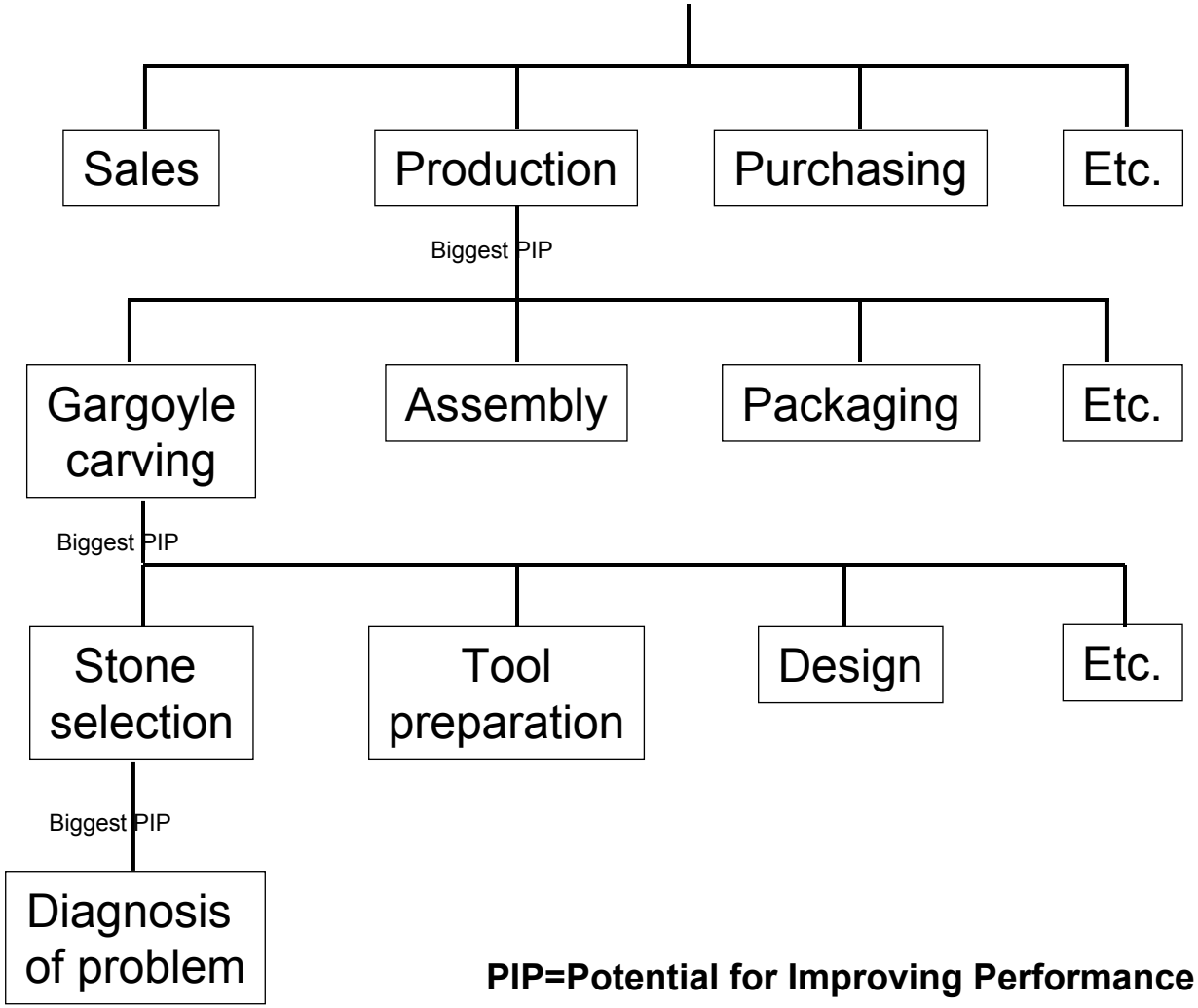


Figure 2.2, p. 60

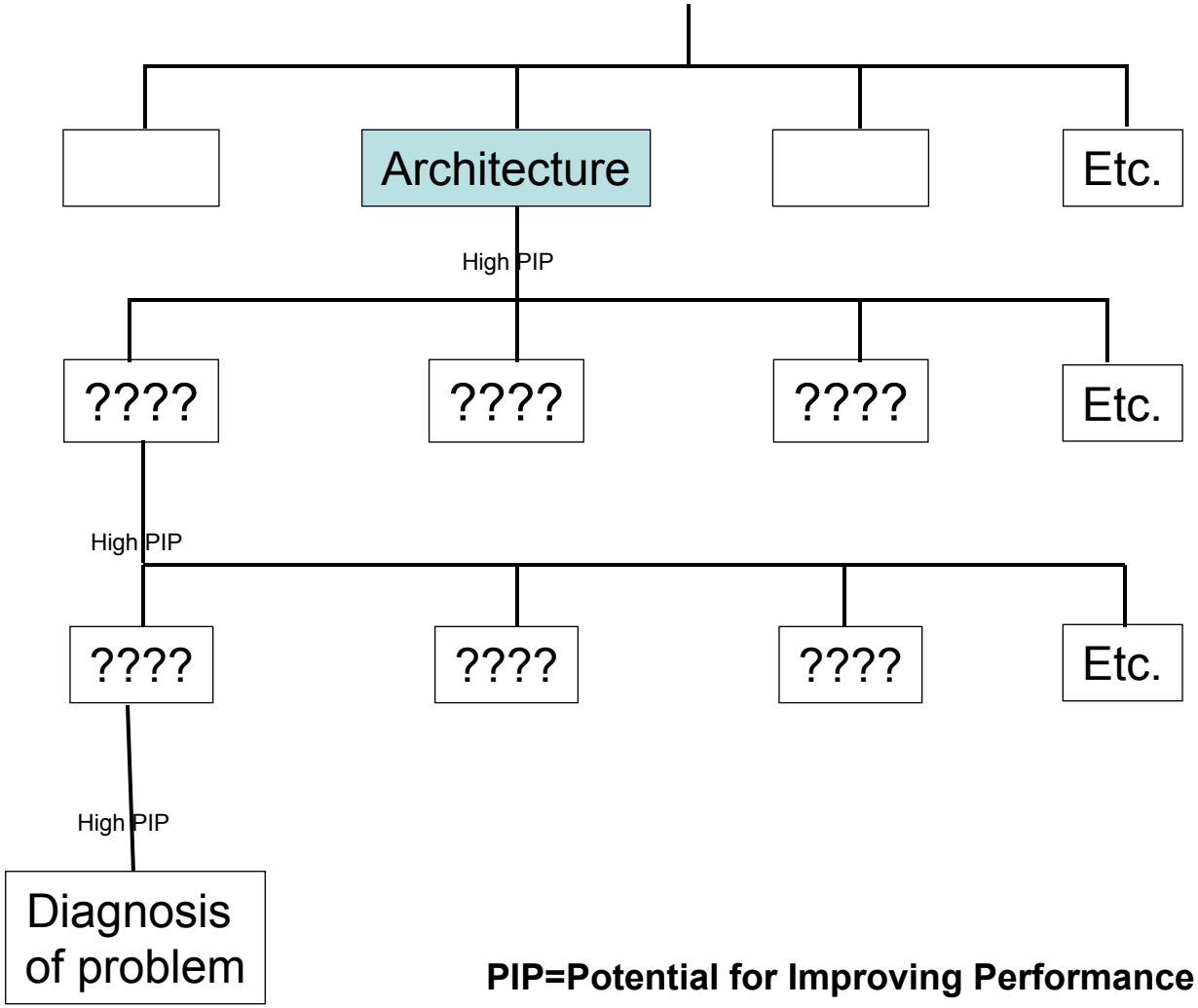
PIP=Potential for Improving Performance

How to apply this to architecture? What goes in the boxes?

First measurements (Institution's accomplishments)

Second measurements (Job accomplishments)

Third measurements (Task accomplishments)



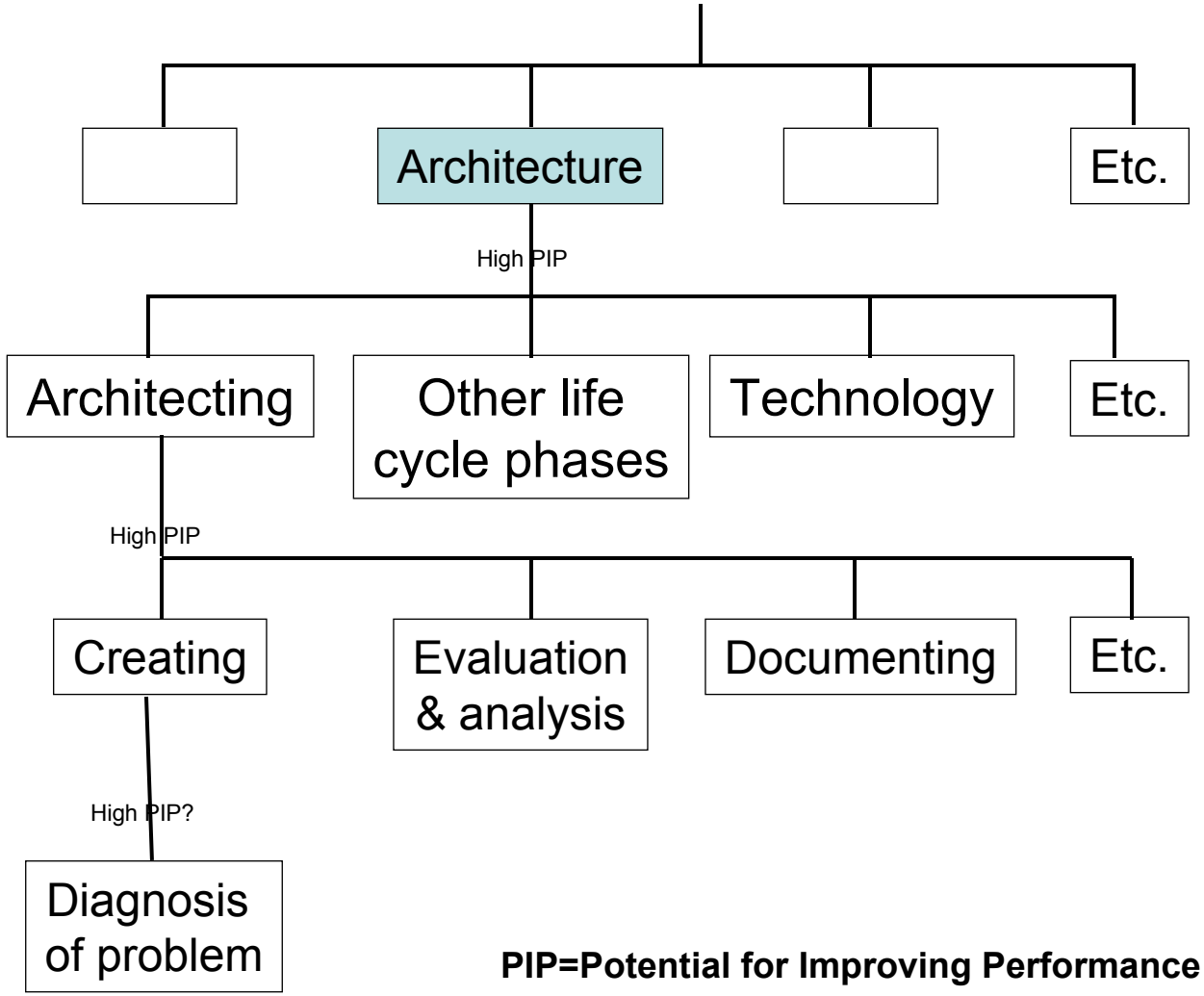
PIP=Potential for Improving Performance

Answer: The architect's duties from our survey

First measurements (Institution's accomplishments)

Second measurements (Job accomplishments)

Third measurements (Task accomplishments)



PIP=Potential for Improving Performance

Game plan to build competence model based on human performance

1. Identify what "worthy performance" means for each task involved in architecture.
2. Identify what costs are involved for each task involved in architecture.
3. Identify performance-related measures of each
4. Identify an exemplary measure – the best we could hope for – of each
5. Build an assessment instrument that will gather measurements in an organization, compare them to exemplar in each category, and identify best potential areas for performance improvement.

Once we've identified an area for improvement, we will still have to suggest specific improvement strategies. (For us, this is farther in the future.)

Sneak Peek: Gilbert's Behavior Model Showing Improvement Opportunities

Generalized description of Behavior Engineering Model, showing the things we can do to increase competence through greater behavior efficiency:

-----E: Environment Supports ----------

Data	*Instruments*	*Incentives*
Relevant/frequent feedback about performance adequacy; Descriptions of expected performance. Clear and relevant guides to adequate performance.	Tools and materials designed to match human factors.	Adequate financial incentives contingent upon performance; Non-monetary incentives;

------P: Person's Repertory of Behavior ----------

Knowledge	*Capacity*	*Motives*
Training matching exemplary performance; Placement	Flexible scheduling to match peak capacity; Prosthesis; Shaping; Adaptation; Selection.	Assessment of motives; Recruitment of people to match realities of situation.

Game plan to build competence model based on human performance: First steps

1. Identify what "worthy performance" means for each task involved in architecture.
2. Identify what costs are involved for each task involved in architecture.
3. Identify performance-related measures of each
4. Identify an exemplary measure – the best we could hope for – of each
5. Build an assessment instrument that will gather measurements in an organization, compare them to exemplar in each category, and identify best potential areas for performance improvement.

Once we've identified an area for improvement, we will still have to suggest specific improvement strategies. (For us, this is farther in the future.)

Example: Creating the architecture

Quality

- Accuracy: Is the architecture the right one for the task at hand?
  Measure: Total cost of changes (= revisiting decisions) to the architecture during development [accounts for lots of small changes as well as number of big ones]. Cost means cost of making change in the architecture AND cost of downstream resulting changes. Measure as % of total cost of system, to (a) find exemplar; and (b) compare systems.
  Comments: Doesn't help for changes that were too expensive to address. Alternative measure is to capture satisfaction of important requirements (e.g., QA scenarios) and test fulfillment (e.g., ATAM-style walkthroughs).
- Class: How many architectures were influenced by this one? Whole thing? Pieces? Ideas?
- Novelty: N/A

Quantity (or Productivity)

- Rate: Time to completion.
- Timeliness: Deadlines met.
- Volume: Size of system.

Cost

- Labor (behavior repositories): Count staff hours for architects
- Material (environmental support): Staff hours for consultants; costs of tools used by architect. Travel costs. Communication costs.
- Management: Count staff hours for managers

Example: Architecture evaluation/analysis

Quality

- Accuracy: Did you do it? Did you miss any problems that came up later, and what were their severities? Measure: cost of changes to the architecture and downstream changes that went undetected by the analysis.
- Class: n/a
- Novelty: n/a

Quantity (or Productivity)

- Rate: Time to completion.
- Timeliness: Meet deadlines.
- Volume: How many different analyses were done?

Cost

- Labor (behavior repositories): Staff hours
- Material (environmental support): Tools for evaluation/analysis, maybe consultants
- Management: Staff hours

Normalize all measures by system size.

Comments: These assume a well-defined evaluation process or analysis task (e.g., build an RMA model, conduct an ATAM exercise).

Example: Architecture documentation

Quality

- Accuracy: Errors discovered in the documentation. Measure how (much) people used (relied on) the documentation. How many times has it been referred to?
- Class: Utility of documentation. Read it and rate it.
- Novelty: n/a

Quantity (or Productivity)

- Rate: Time to completion.
- Timeliness: Deadlines.
- Volume: "Right" size? 80-120 pages? Delta from ideal.

Cost (Same as others)

Agenda

Part 1. Introduction

Part 2. Duties, skills, and knowledge of software architects

Part 3. A human performance model for architecture competence

Part 4. Conclusions

Architecture Competence Project

Our mission is to help individuals, teams, and organizations measure and improve their competence in architecture.

We are pursuing two major themes:

- Human performance model, informed by duties/skills, and knowledge
- Organizational coordination mechanisms

The work also involves

- Exploring the relationship between business goals and quality attributes
- Surveying community about best practices for architects and organizations
- Crafting pilot assessment instruments
- Pursuing case studies in competence improvement

Conclusions

Gilbert gives us operational steps to prosecute architecture competence

- Define worthy performance in tasks and sub-tasks of architecture
- Define measures
- Define exemplary performance
- Gather measurements in an organization
- Find areas of most potential improvement
- Identify strategies for improvement

This gives us a model for architecture competence that has a solid pedigree.

Our duties/skills/knowledge survey directly supports the model

- Duties become task areas
- Skills and knowledge are improvement areas

Coming attractions

May: Working session at the SEI's Software Architecture Technology User Network (SATURN) conference, 14-17 May, Pittsburgh.

You're invited! Visit www.sei.cmu.edu/saturn

June: Writing an overview Technical Note

Ongoing: Questionnaires for practicing architects

4 pages, less than 30 minutes

You're invited! If you'd like to fill out a questionnaire, send me e-mail:

clements@sei.cmu.edu

Acknowledgments

Members of the SEI Architecture Competence Project

- Len Bass
- Paul Clements
- Rick Kazman
- Mark Klein

The duties/skills/knowledge survey was carried out by

- Prageti Verma, Symbosis Centre for Information Technology, Pune
- Shivani K. Reddy, Symbiosis Centre for Information Technology, Pune
- Divya Devesh, Indian Institute of Technology Guwahati

Read more at www.sei.cmu.edu/architecture