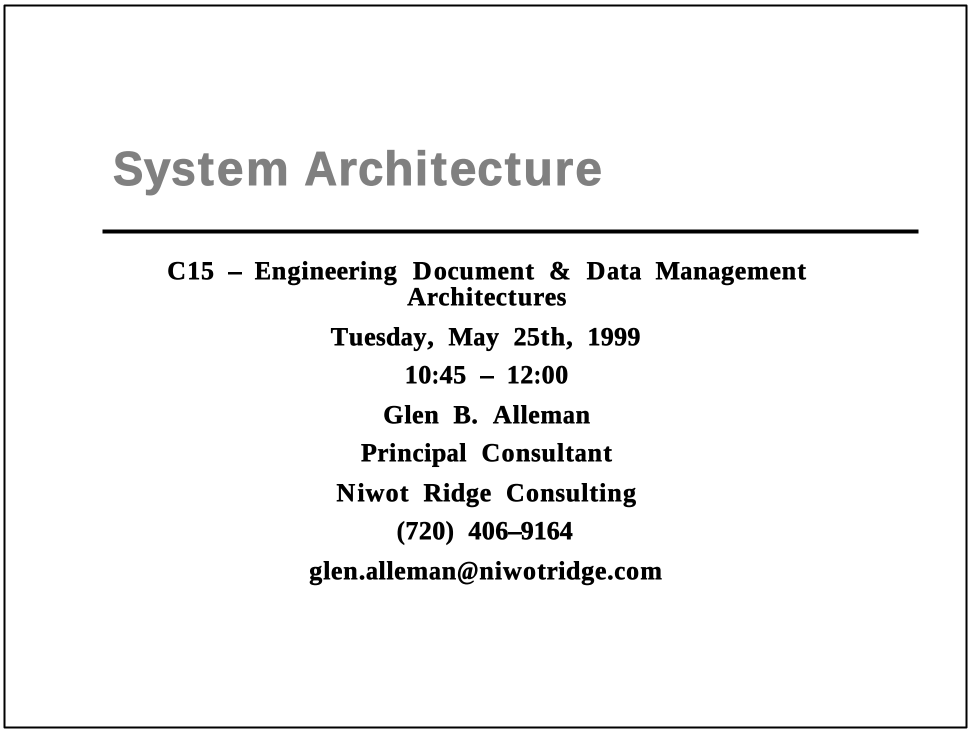


System Architecture

C15 – Engineering Document & Data Management Architectures

Tuesday, May 25th, 1999

10:45 – 12:00

Glen B. Alleman

Principal Consultant

Niwot Ridge Consulting

(720) 406–9164

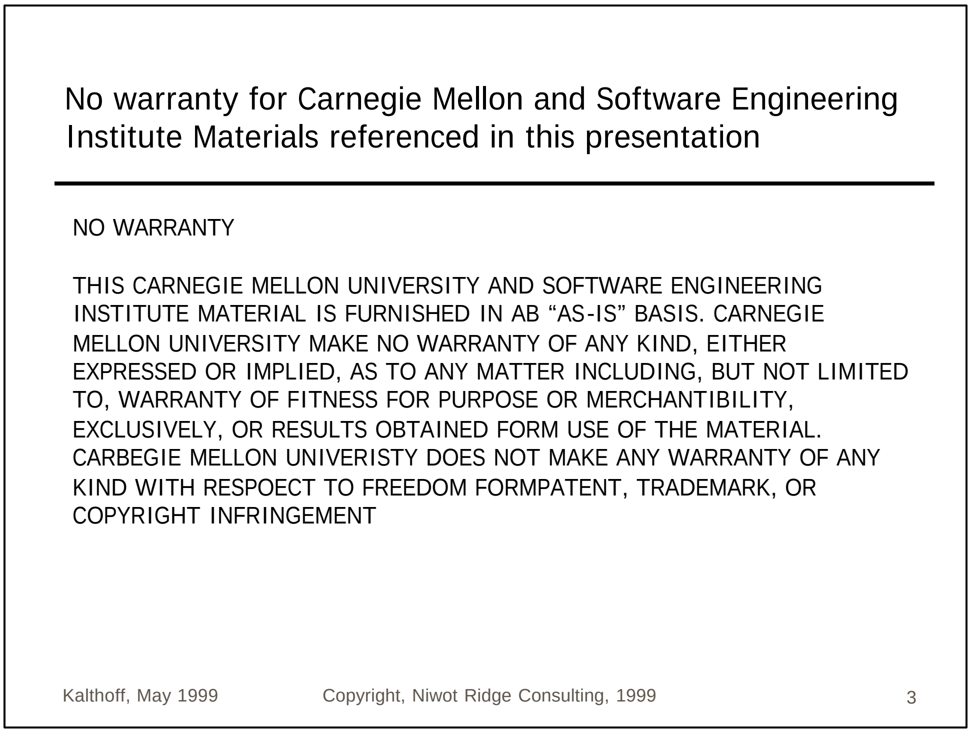
glen.alleman@niwotridge.com

- This is the C15 session titled System Architecture.
- We'll discuss system architecture and how it relates to EDM / PDM and ERP systems as well as how to set up an Architecture Review Board.

System Architecture

How good system architecture can be used to improve the chances of success for an EDM/PDM/ERP project.

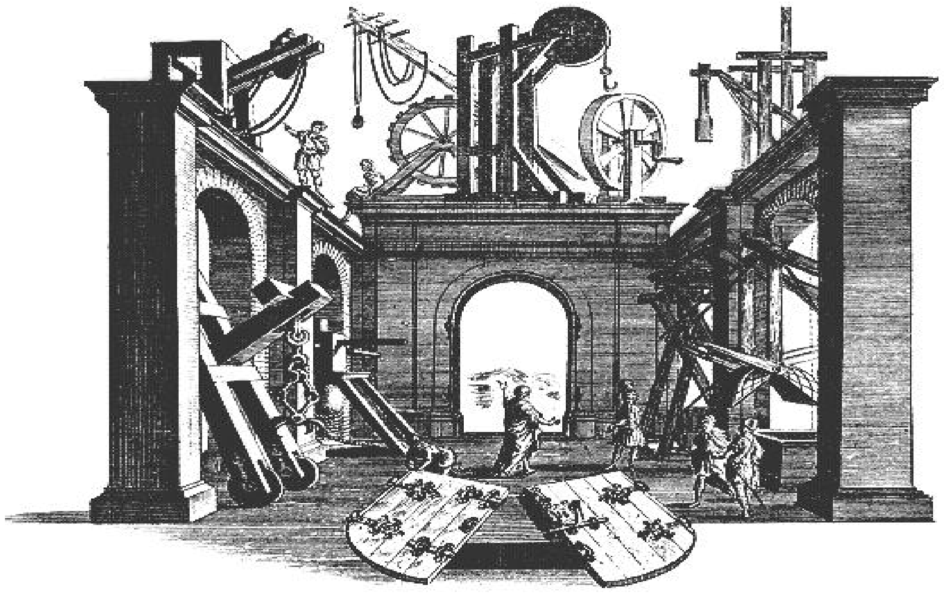
- We’re going to talk about architecture this morning and how architecture can provide tools to help a project survive.
- This may appear to be an esoteric subject, one not normally found at a conference like this, with its emphasis on buyer and vendor relationships.
- I hope you will leave this session with a better understanding of how architecture fits into the process of acquiring systems. And how architecture is the foundation of a successful system deployment.



No warranty for Carnegie Mellon and Software Engineering Institute Materials referenced in this presentation

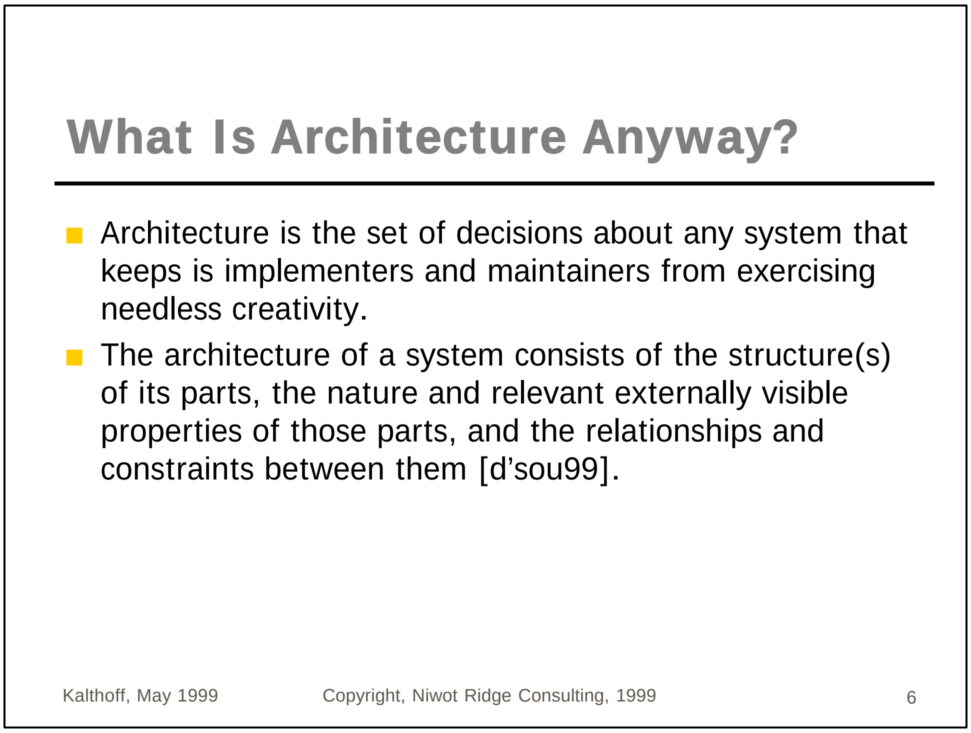
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 3

- Before we get started, here’s the legal mumbo jumbo for the materials presented here.



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 5

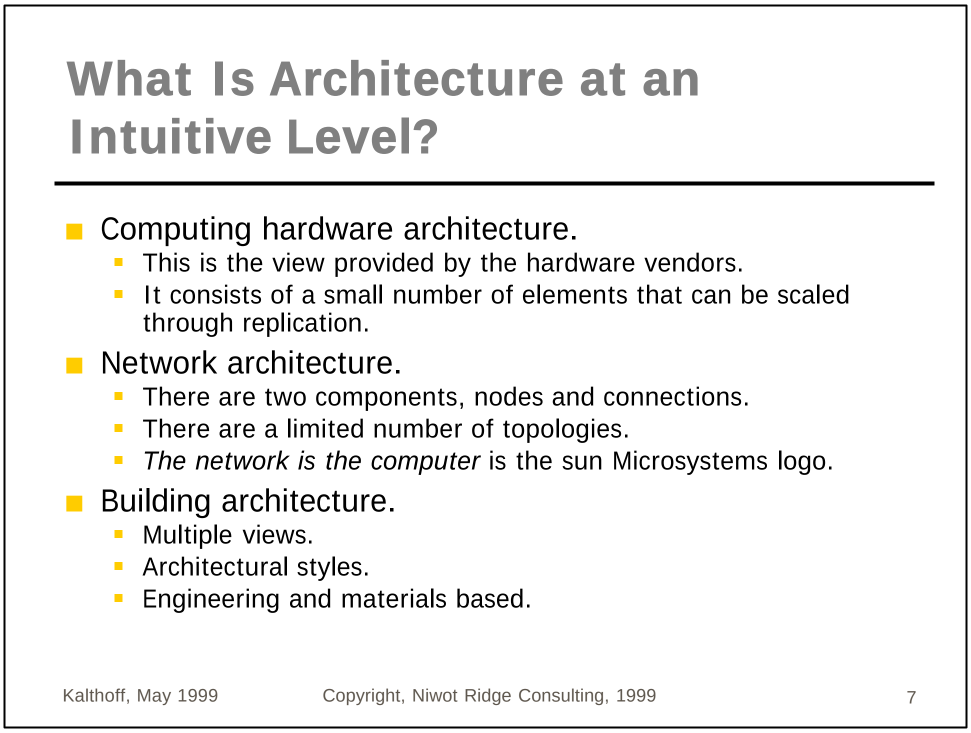
- Is this what you think of when someone mentions your IT system's architecture?



What Is Architecture Anyway?

- Architecture is the set of decisions about any system that keeps is implementers and maintainers from exercising needless creativity.
- The architecture of a system consists of the structure(s) of its parts, the nature and relevant externally visible properties of those parts, and the relationships and constraints between them [d'sou99].

- When I use the word *architecture* I mean it in the context of software systems. Here are some definitions found in the literature. These are not *official* definitions (there are none), but ones that convey the meaning of today's talk.

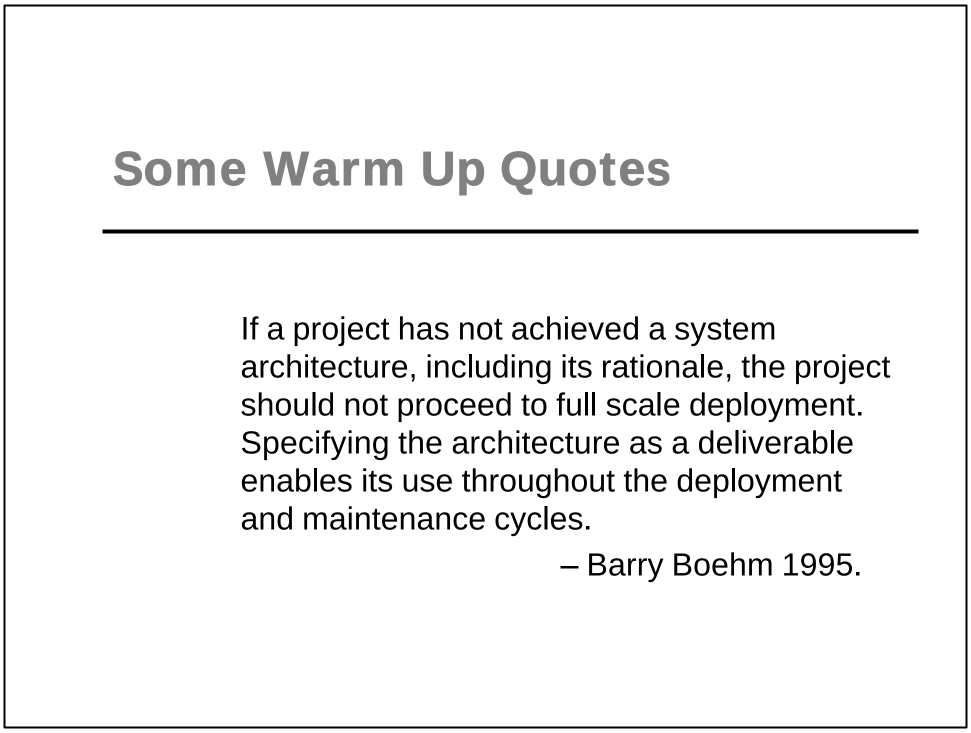


What Is Architecture at an Intuitive Level?

- Computing hardware architecture.
  - This is the view provided by the hardware vendors.
  - It consists of a small number of elements that can be scaled through replication.
- Network architecture.
  - There are two components, nodes and connections.
  - There are a limited number of topologies.
  - *The network is the computer* is the sun Microsystems logo.
- Building architecture.
  - Multiple views.
  - Architectural styles.
  - Engineering and materials based.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 7

- How can architecture be put is some context. Everyone talks about architecture, but the meanings are always different depending on the point of view or on what is being sold.
- The hardware vendors talk about their architecture. The network vendors talk about their architecture. The building architecture talk about their architecture.
- Each of these descriptions provides very little information.
- In the end, because no one talks about architecture in the context of business problems.

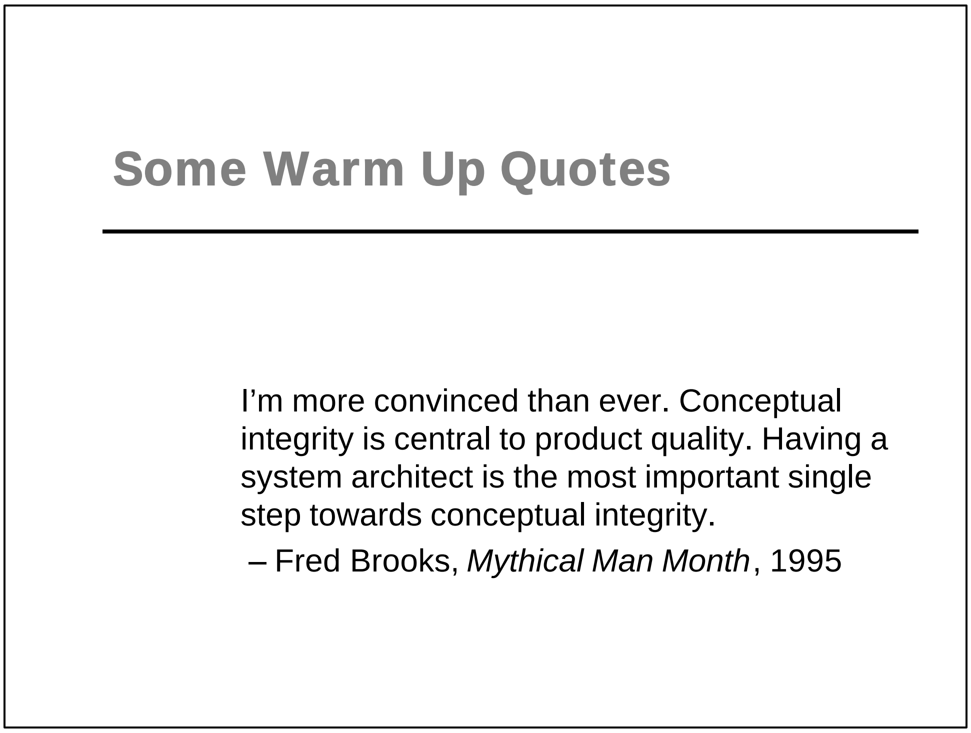


Some Warm Up Quotes

If a project has not achieved a system architecture, including its rationale, the project should not proceed to full scale deployment. Specifying the architecture as a deliverable enables its use throughout the deployment and maintenance cycles.

– Barry Boehm 1995.

- Barry Boehem is currently a professor of Computer Science at the University of Southern California.
- Prior to that he had a distinguished career at TRW and the Software Engineering Institute.
- Barry lead the effort at TRW to develop software engineering as a profession.
- Barry’s quotation does not come from academic experience, but from decades of leading real time software development efforts, ranging from ABM software to state of the art communications systems.

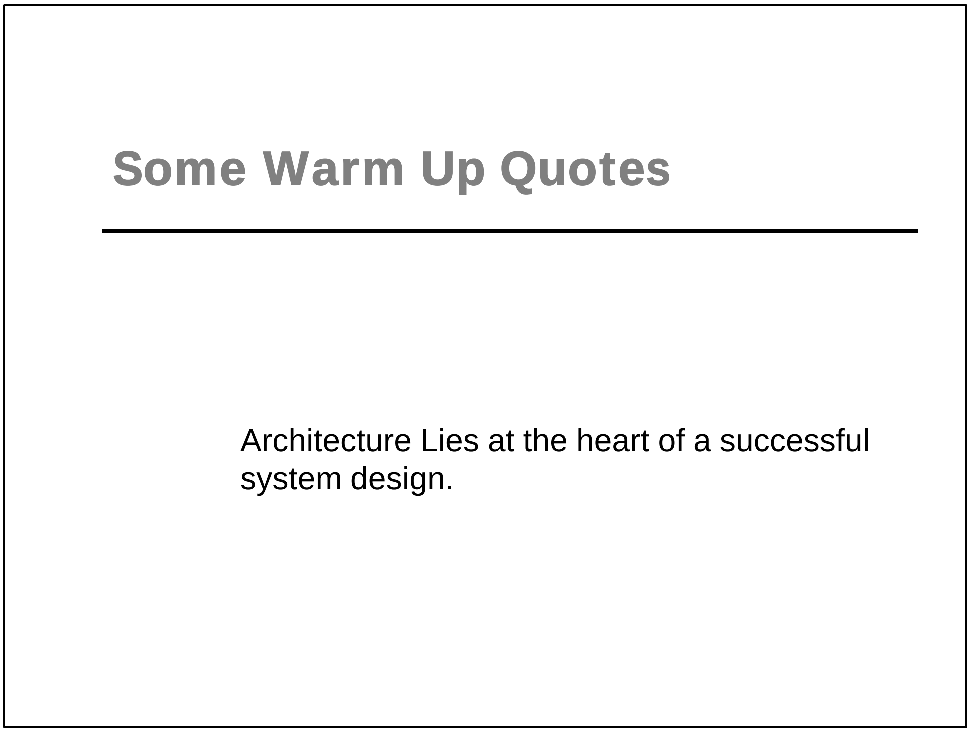


Some Warm Up Quotes

I’m more convinced than ever. Conceptual integrity is central to product quality. Having a system architect is the most important single step towards conceptual integrity.

– Fred Brooks, *Mythical Man Month*, 1995

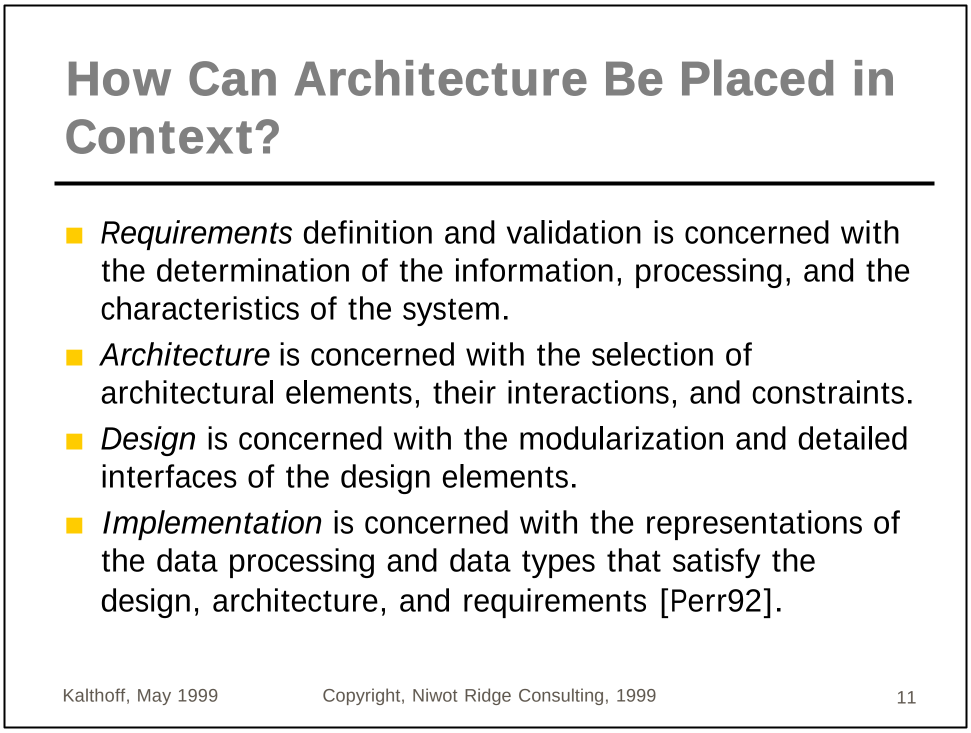
- Fred Brooks is the father of OS/360.
- Fred’s book *The Mythical Man Month* is mandatory reading for anyone hoping to lead a software system project.



Some Warm Up Quotes

Architecture Lies at the heart of a successful system design.

- This is a core understanding for the seminar session today.
- If you currently don’t accept this statement, hopefully you will at the end of the session.

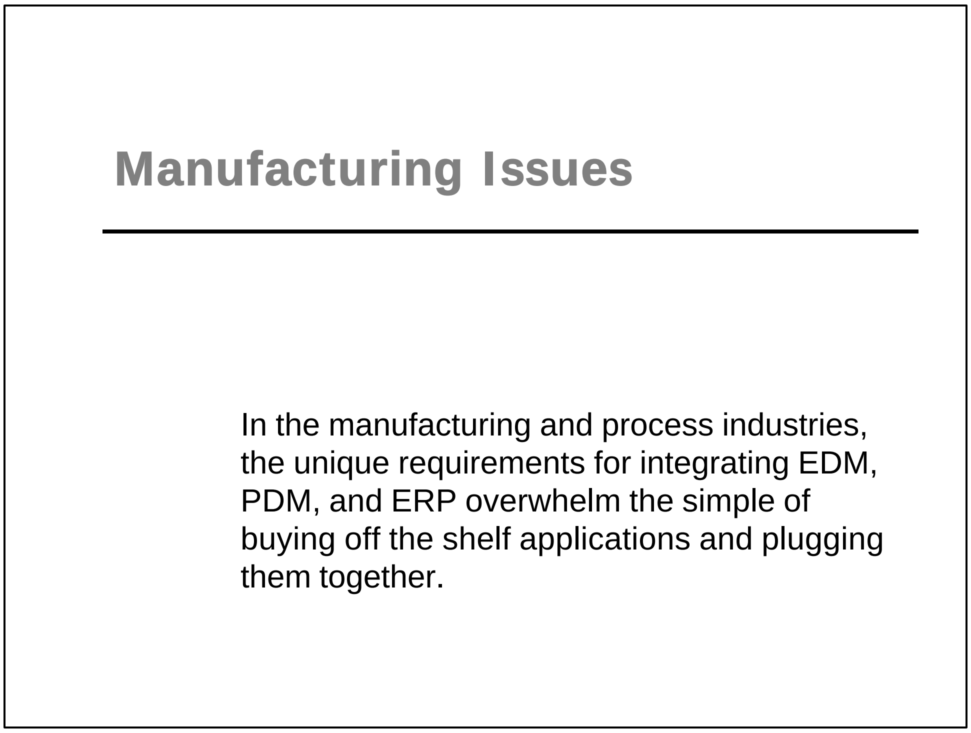


How Can Architecture Be Placed in Context?

- *Requirements* definition and validation is concerned with the determination of the information, processing, and the characteristics of the system.
- *Architecture* is concerned with the selection of architectural elements, their interactions, and constraints.
- *Design* is concerned with the modularization and detailed interfaces of the design elements.
- *Implementation* is concerned with the representations of the data processing and data types that satisfy the design, architecture, and requirements [Perr92].

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 11

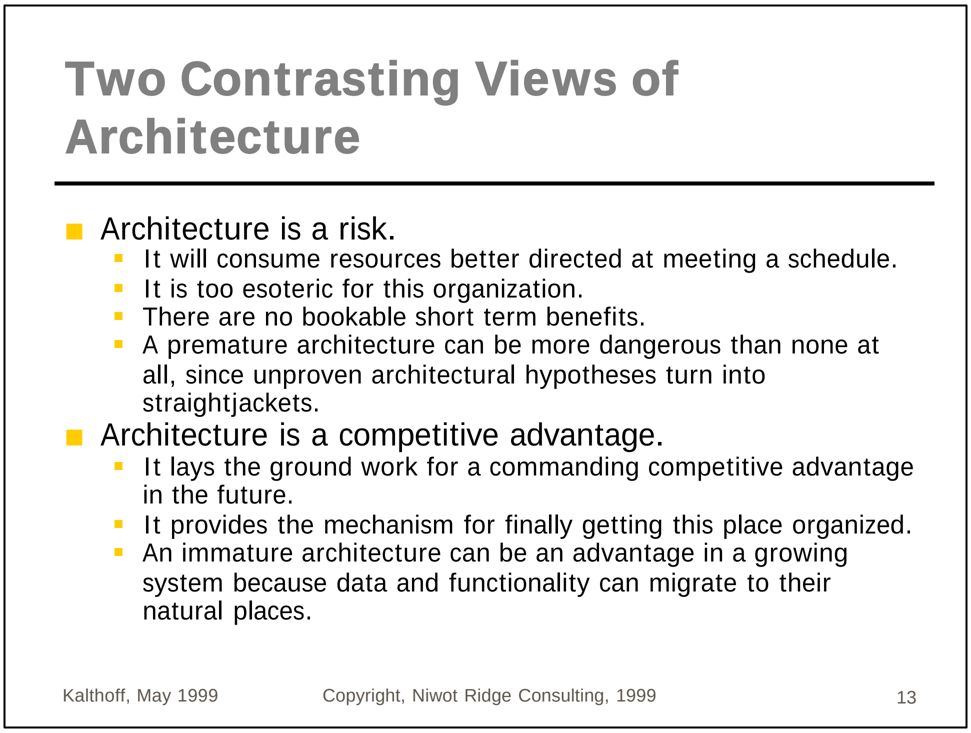
- Architecture is related to requirements, design, and implementation. Each of these activities cannot function without the others.
- Starting with requirements, the users, buyers, operators, maintainers and owners of the system must specify their respective requirements. The capturing of these requirements is a discipline unto itself.
- Depending on the complexity of the system, as seen from any of the stakeholder views, the requirements are simple or complex.
- Without architecture, the other components don’t hold together.
  - The ball of Mud analogies describe how existing systems work, but don’t have architecture.



Manufacturing Issues

In the manufacturing and process industries, the unique requirements for integrating EDM, PDM, and ERP overwhelm the simple of buying off the shelf applications and plugging them together.

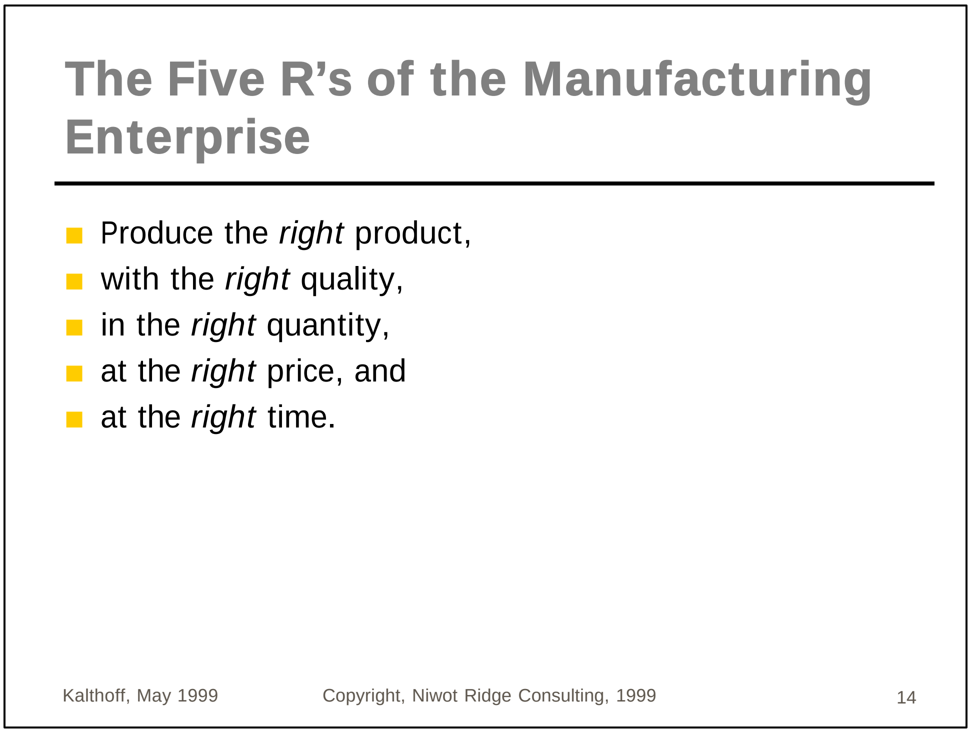
- We should be able to convince ourselves that the manufacturing domain is different than in insurance or banking.
- There are multiple versions of the same part, product, or document.
- There are dynamic relationships between objects.



Two Contrasting Views of Architecture

- Architecture is a risk.
  - It will consume resources better directed at meeting a schedule.
  - It is too esoteric for this organization.
  - There are no bookable short term benefits.
  - A premature architecture can be more dangerous than none at all, since unproven architectural hypotheses turn into straightjackets.
- Architecture is a competitive advantage.
  - It lays the ground work for a commanding competitive advantage in the future.
  - It provides the mechanism for finally getting this place organized.
  - An immature architecture can be an advantage in a growing system because data and functionality can migrate to their natural places.

- Before we get too far into the time allotted, let’s take a short time out.
- The concepts we’re going to be talking about have no black or white answers. All the discussion is built on subjective materials. There can be systems built with no consideration to architecture that work fine, last a long time and are cheap.
- There can be systems that have wonderful architecture and are complete failures.

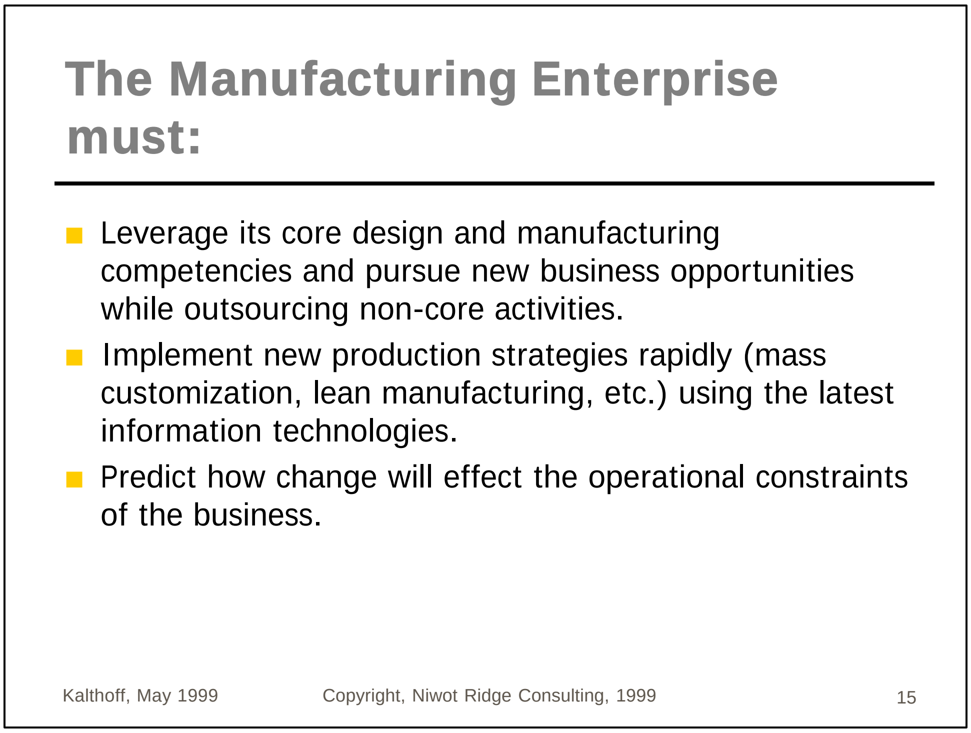


The Five R's of the Manufacturing Enterprise

- Produce the *right* product,
- with the *right* quality,
- in the *right* quantity,
- at the *right* price, and
- at the *right* time.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 14

- In order to stay focused on the manufacturing paradigm, here's some office poster material.
- Our motivation for the remaining materials will be to discover how architecture can support the Five R's.

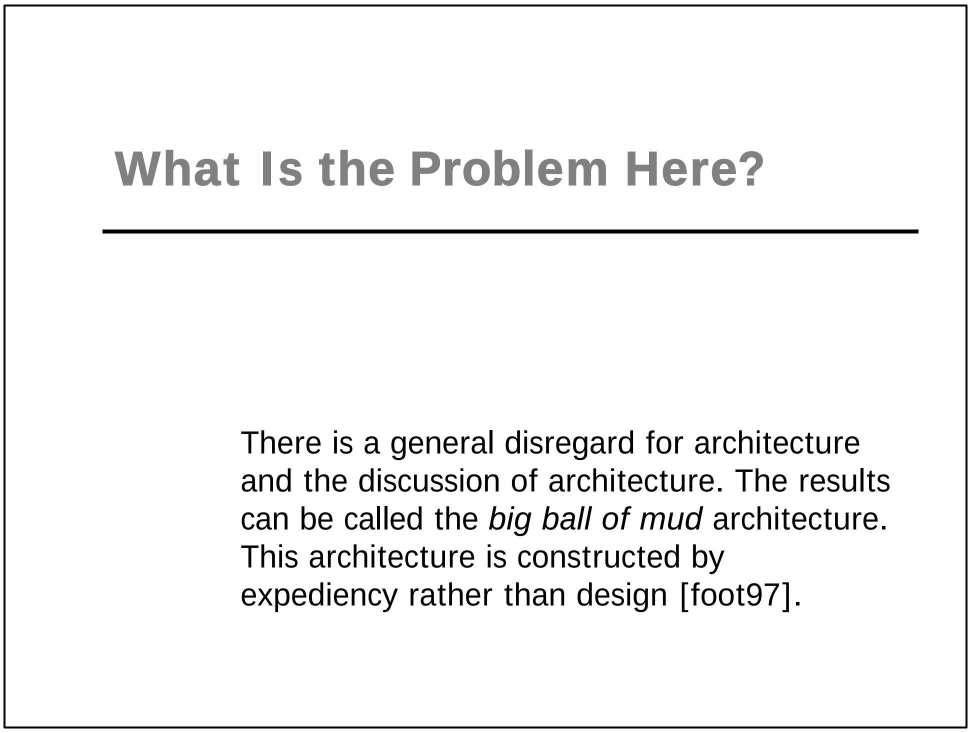


The Manufacturing Enterprise must:

- Leverage its core design and manufacturing competencies and pursue new business opportunities while outsourcing non-core activities.
- Implement new production strategies rapidly (mass customization, lean manufacturing, etc.) using the latest information technologies.
- Predict how change will effect the operational constraints of the business.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 15

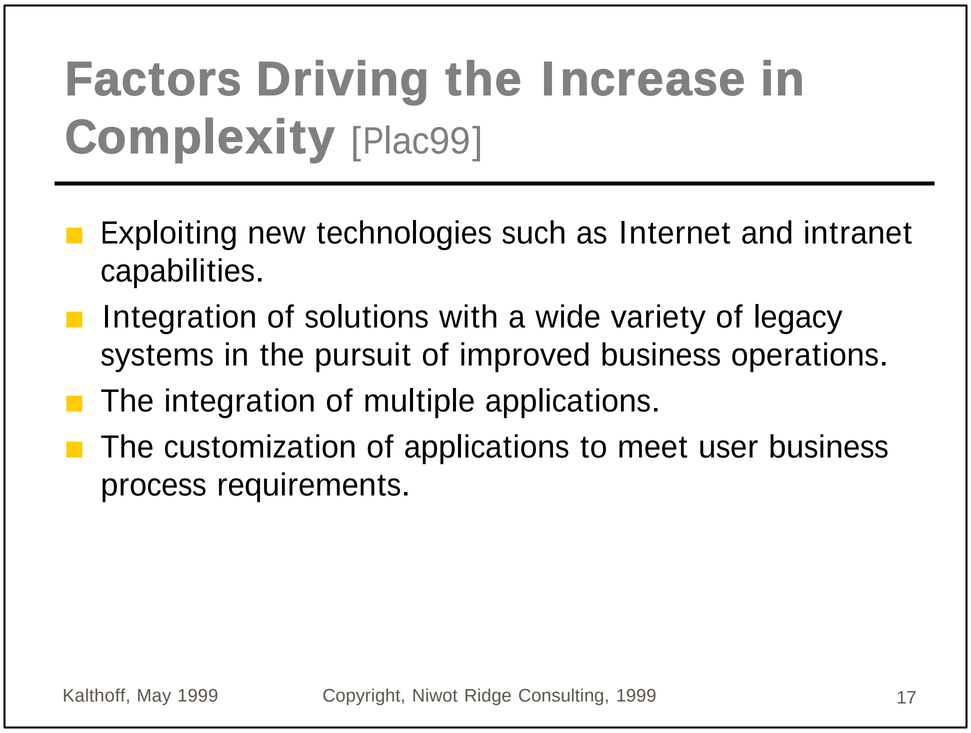
- The steps taken in pursuit of an architecture-based design process should be based on the following.



What Is the Problem Here?

There is a general disregard for architecture and the discussion of architecture. The results can be called the *big ball of mud* architecture. This architecture is constructed by expediency rather than design [foot97].

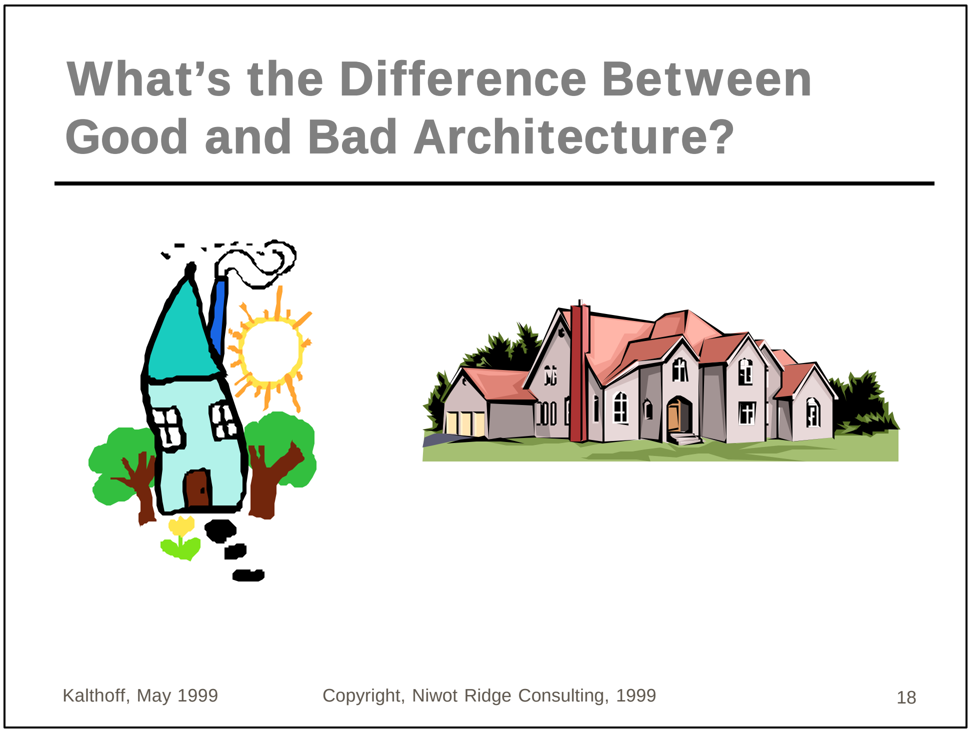
- All this talk of architecture usually gets people all riled up...you’re just wasting my time with all this fancy talk.
- Architecture has nothing to do with MY problem.
- Let’s get this system installed and everyone can get back to work.
- Oh, I forgot to tell you one last thing the system has to do...


Factors Driving the Increase in Complexity [Plac99]

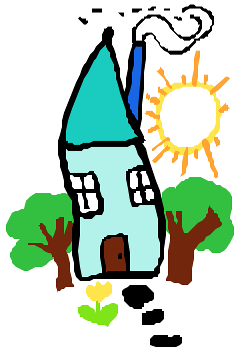
- Exploiting new technologies such as Internet and intranet capabilities.
- Integration of solutions with a wide variety of legacy systems in the pursuit of improved business operations.
- The integration of multiple applications.
- The customization of applications to meet user business process requirements.

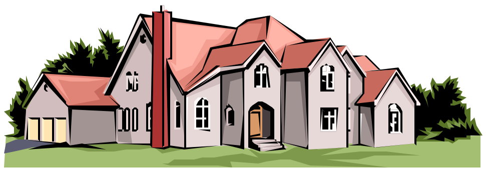
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 17

- The primary motivator for complexity is the WEB.
- Ad to that the integration of legacy applications and we have more than just an integration problem.
- The issue here is one of merging paradigms, the web and traditional systems.



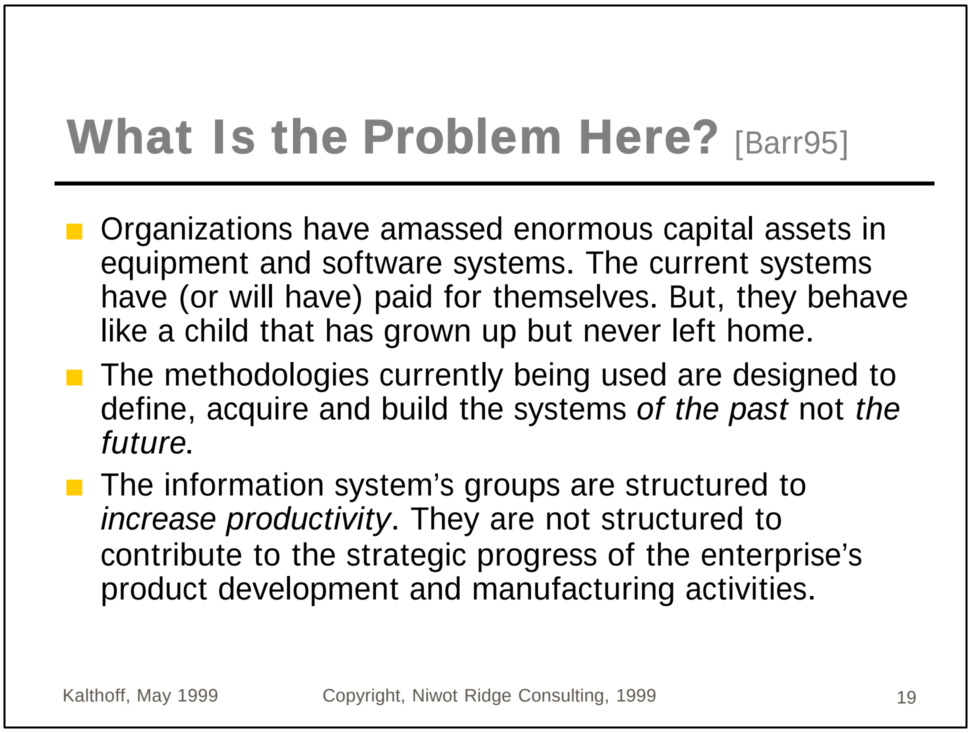
What's the Difference Between Good and Bad Architecture?





Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 18

- We all have some experience in designing a home project, a dog house, a bird house, maybe even a real house.
- Hopefully from that experience, we got some understanding of what an architect does for a living.
- Even though there are plenty of *badly* architected buildings and houses around, the profession of architecture is something that requires training and experience and most of all talent.

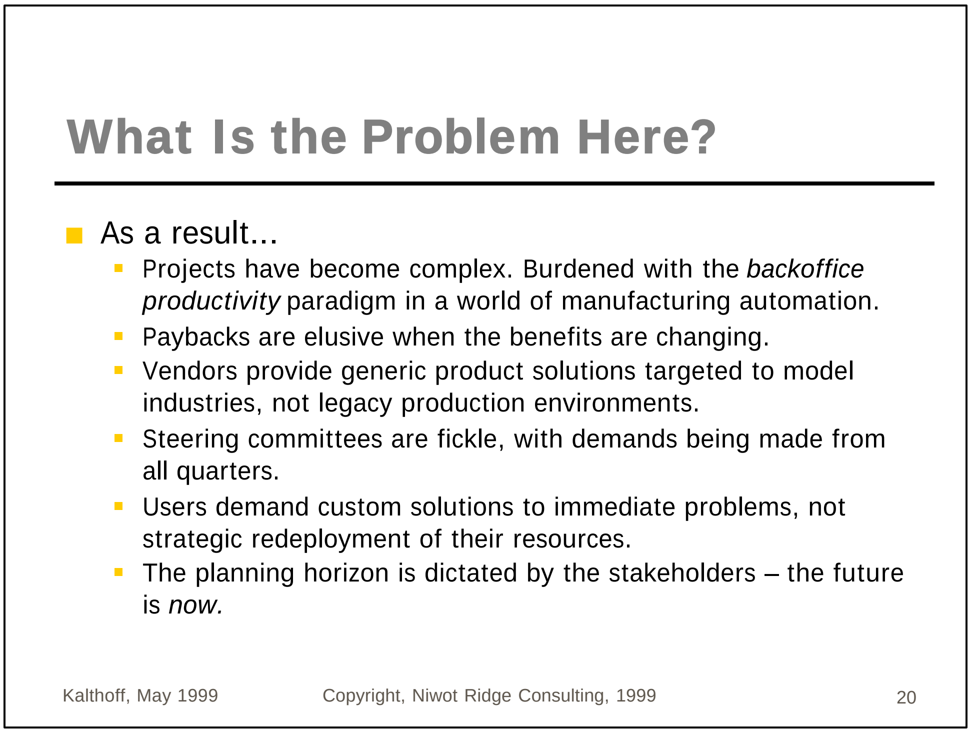


What Is the Problem Here? [Barr95]

- Organizations have amassed enormous capital assets in equipment and software systems. The current systems have (or will have) paid for themselves. But, they behave like a child that has grown up but never left home.
- The methodologies currently being used are designed to define, acquire and build the systems *of the past* not *the future*.
- The information system's groups are structured to *increase productivity*. They are not structured to contribute to the strategic progress of the enterprise's product development and manufacturing activities.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 19

- There has been lots of research on why systems don't work, why systems cost too much, why the organizations that manage these systems fall apart.
- But there are some core issues that will make it clear (hopefully) to everyone here.
- These are issues having to do with how we manage the process and how we see ourselves while performing these management functions.

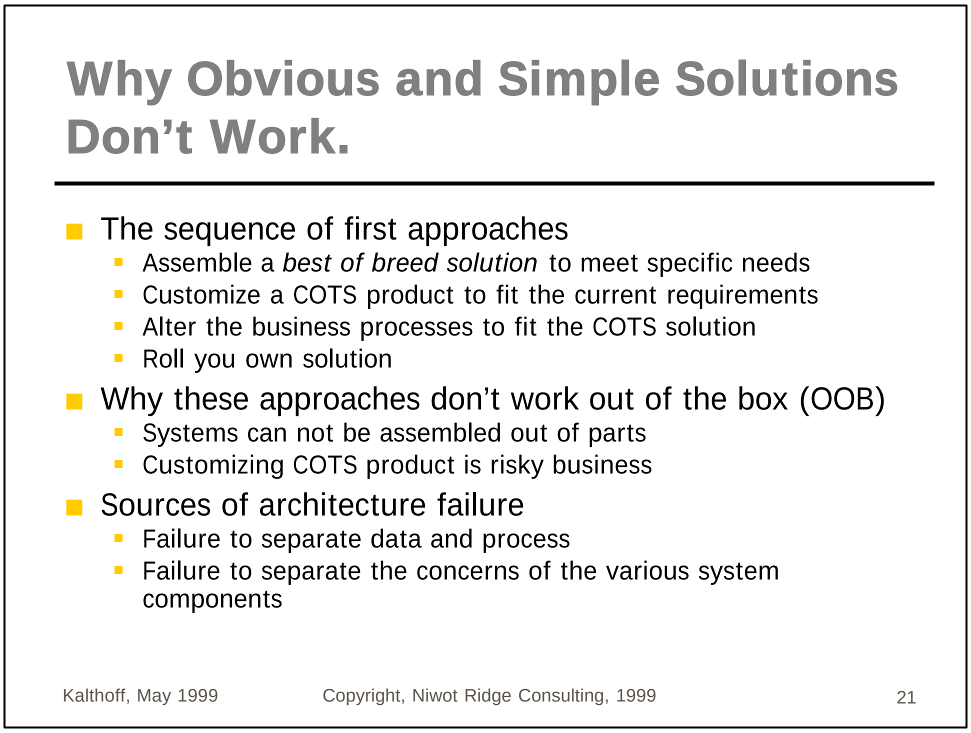


What Is the Problem Here?

- As a result…
  - Projects have become complex. Burdened with the *backoffice productivity* paradigm in a world of manufacturing automation.
  - Paybacks are elusive when the benefits are changing.
  - Vendors provide generic product solutions targeted to model industries, not legacy production environments.
  - Steering committees are fickle, with demands being made from all quarters.
  - Users demand custom solutions to immediate problems, not strategic redeployment of their resources.
  - The planning horizon is dictated by the stakeholders – the future is *now.*

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 20

- If you accept for the moment the concepts presented in the previous slide, then the results are given here.
- We have all experienced at least one if not more than one of these consequences.

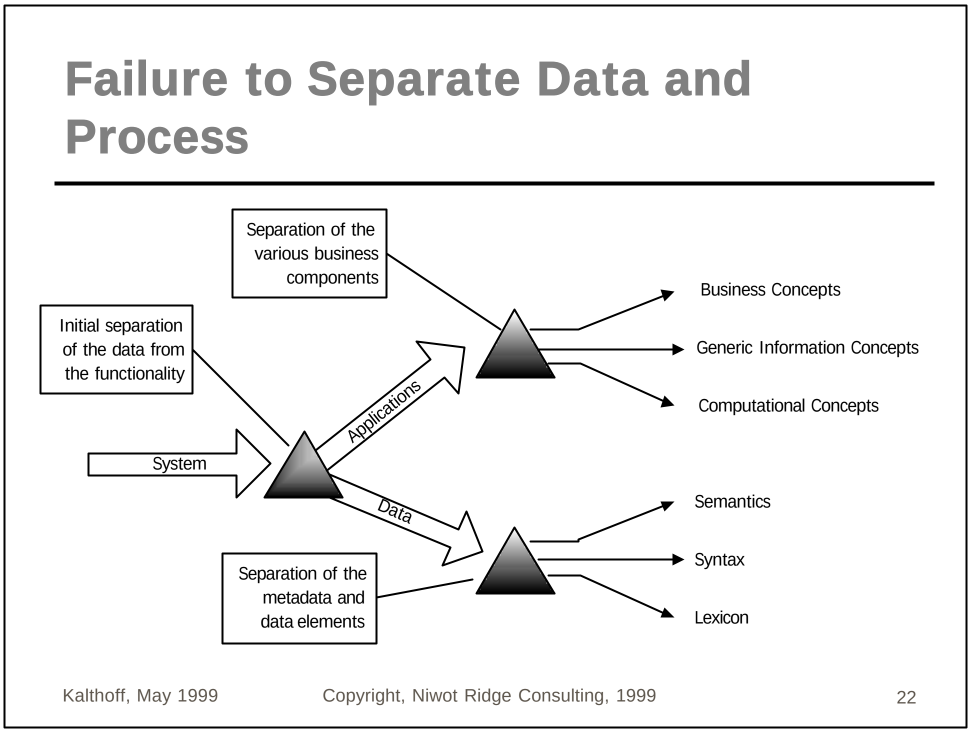


Why Obvious and Simple Solutions Don't Work.

- The sequence of first approaches
  - Assemble a *best of breed solution* to meet specific needs
  - Customize a COTS product to fit the current requirements
  - Alter the business processes to fit the COTS solution
  - Roll you own solution
- Why these approaches don't work out of the box (OOB)
  - Systems can not be assembled out of parts
  - Customizing COTS product is risky business
- Sources of architecture failure
  - Failure to separate data and process
  - Failure to separate the concerns of the various system components

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 21

- If this was easy anyone could do it.
- If this was easy there would be no need for an architecture review board, an IT steering committee, consultants and all the effort being put into fixing systems after they have been purchased.
- Something else must be going on here.
- The truth is *all the easy problems have been solved*.

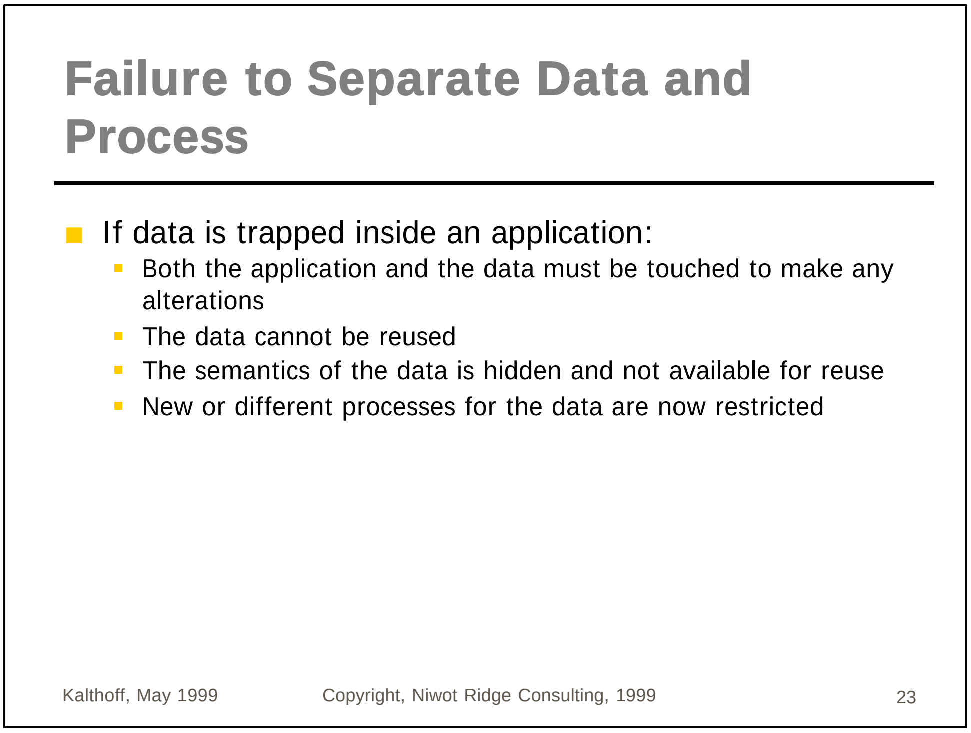


Failure to Separate Data and Process

Separation of the various business components
Initial separation of the data from the functionality
System
Applications
Data
Business Concepts
Generic Information Concepts
Computational Concepts
Semantics
Syntax
Lexicon
Separation of the metadata and data elements

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 22

- In the legacy system integration business, one major source of problems is the failure to separate data and process.
- Here’s a very simple picture of this separation process.
- Volumes have been written and millions invested in this problem...so this is not a trivial issue.

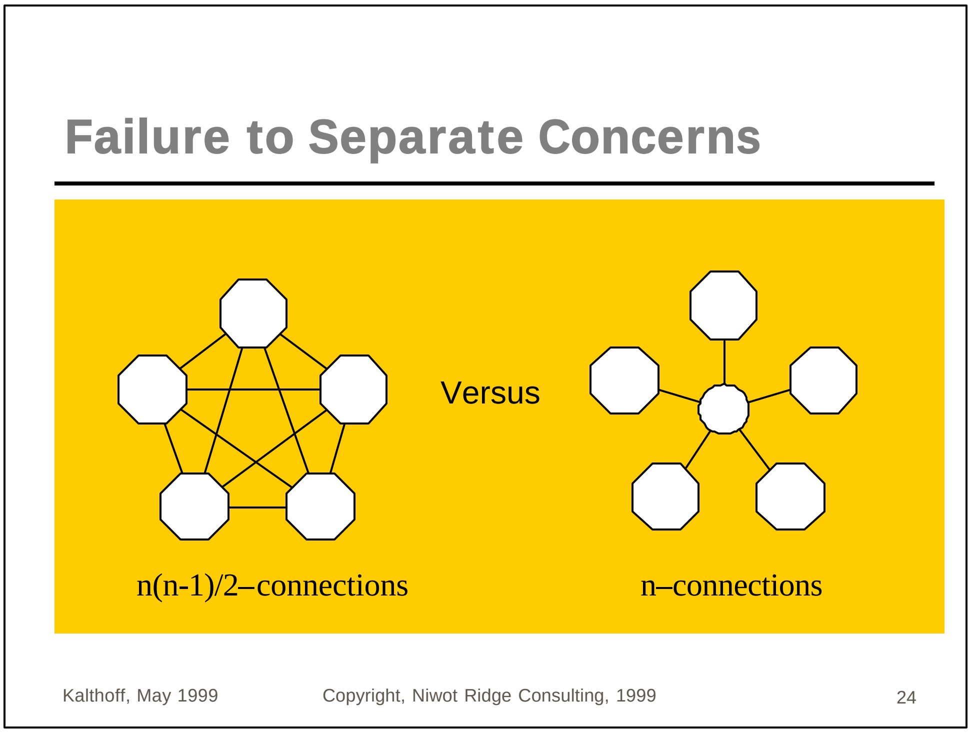


Failure to Separate Data and Process

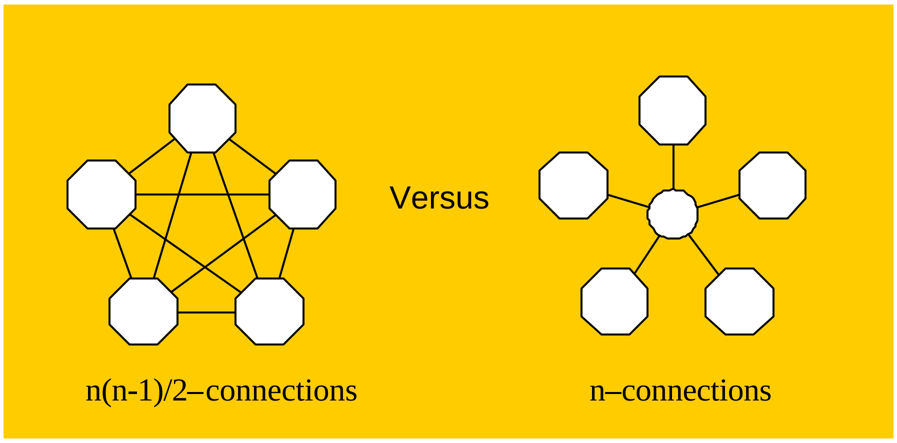
- If data is trapped inside an application:
  - Both the application and the data must be touched to make any alterations
  - The data cannot be reused
  - The semantics of the data is hidden and not available for reuse
  - New or different processes for the data are now restricted

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 23

- The primary failure mode of a poorly architectured system is that the meaning of the data depends on the software application.
- This is more that data typing, units, and measures.
- If data and process are intertwined, there’s going to be trouble.

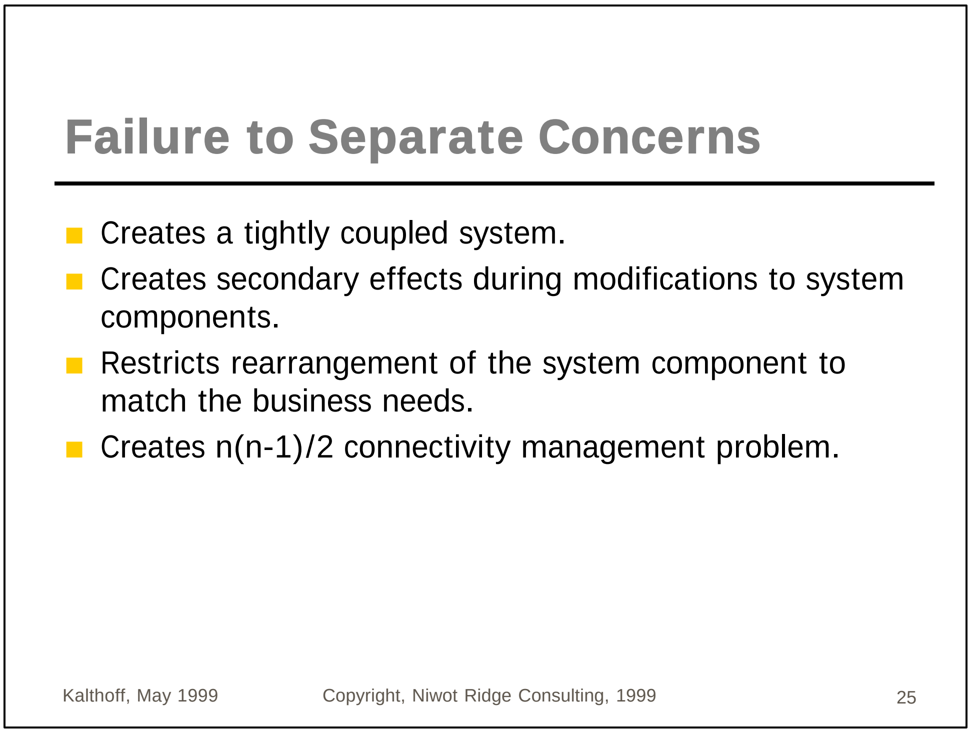


Failure to Separate Concerns



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 24

- Another common architectural failure mode is the intertwining of individual processing and data domains.
- It starts in a simple way, like most big problems in the world.
- “We’ll just take some data from here. Do some simple processing, and put the data there.”
- And then it grows.
- Like the Arab saying about the Camel and the Tent.

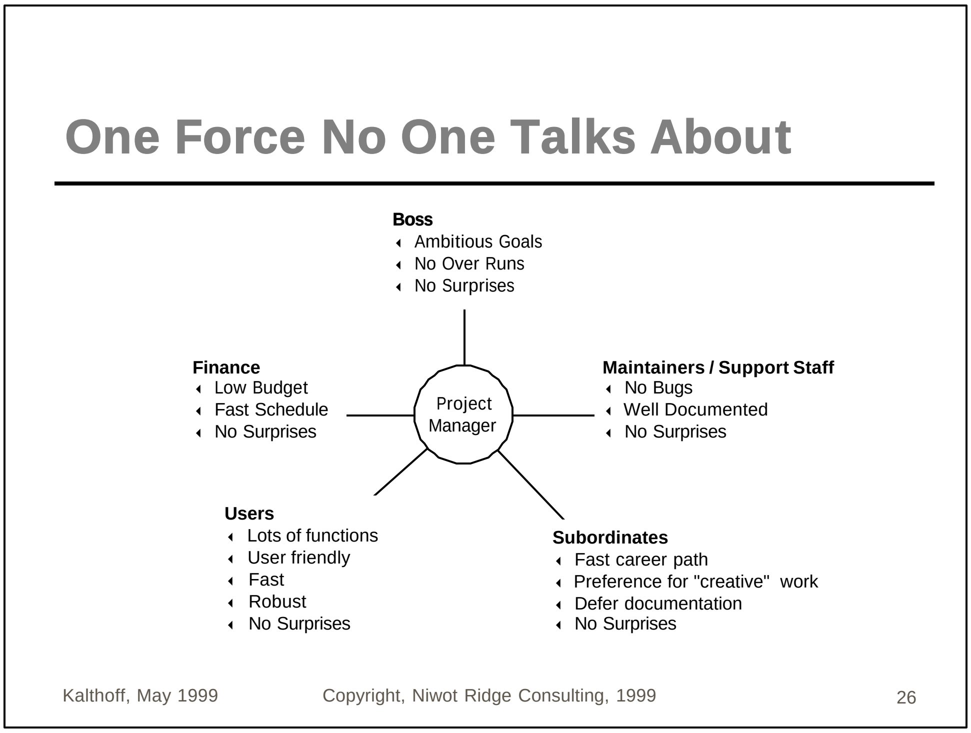


Failure to Separate Concerns

- Creates a tightly coupled system.
- Creates secondary effects during modifications to system components.
- Restricts rearrangement of the system component to match the business needs.
- Creates n(n-1)/2 connectivity management problem.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 25

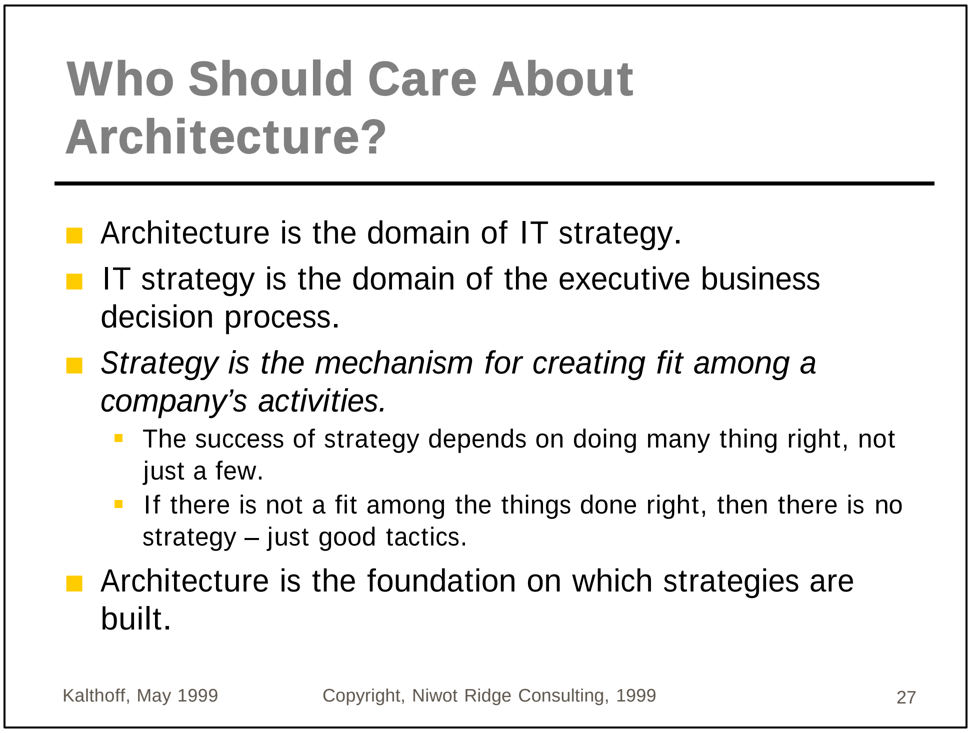
- Once it starts it never ends.
- The small *helper application* you bought becomes a critical business system.
- This small application will have then laced its tentacles into every corner of the system.



One Force No One Talks About

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 26

- Once you decide to do something about the n(n-1)/2 mess.
- Another source of grief are the forces put on the project manager.
- Here’s a simple picture of the major forces. These are simple linear forces.
- In reality these forces are non-linear and interacting.

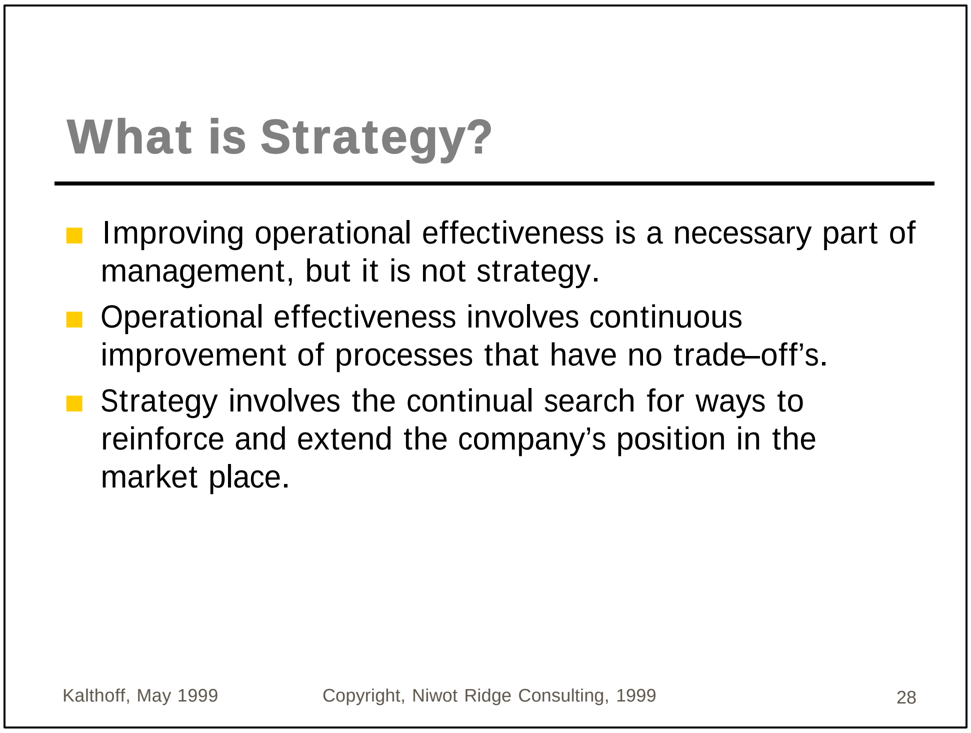


Who Should Care About Architecture?

- Architecture is the domain of IT strategy.
- IT strategy is the domain of the executive business decision process.
- *Strategy is the mechanism for creating fit among a company's activities.*
  - The success of strategy depends on doing many thing right, not just a few.
  - If there is not a fit among the things done right, then there is no strategy – just good tactics.
- Architecture is the foundation on which strategies are built.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 27

- "This is all interesting" you say, but why should I care about this, since I'm here to buy something, or at least look at things that I may want to buy.
- Well, if you have any understanding of building a house or constructing a public space, like a garden or landscaping, you hopefully realize that architecture is an *attribute* of the outcome that conveys style, meaning, feeling, and character.
- In Christopher Alexander's book A Timeless Way of Building, the difference between good and bad architecture is objective, not subjective. Many people disagree with Alexander's view, but Alexander counters this is because the central quality which makes the difference cannot be named.
- This is our problem -- what is that quality?

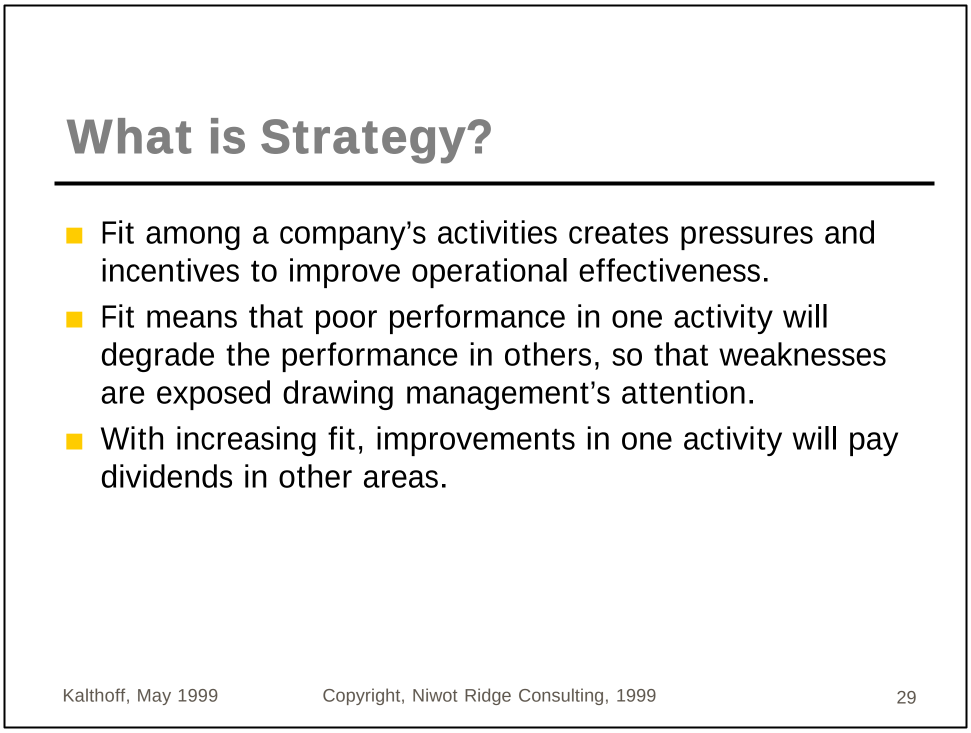


What is Strategy?

- Improving operational effectiveness is a necessary part of management, but it is not strategy.
- Operational effectiveness involves continuous improvement of processes that have no trade–off's.
- Strategy involves the continual search for ways to reinforce and extend the company's position in the market place.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 28

- We've used the word strategy in the previous slide. What is strategy and how is it related to architecture.
- There is confusion already between strategy, architecture, tactics, operational plans, etc.
- Here are the differences, at least for this presentation, between these concepts.

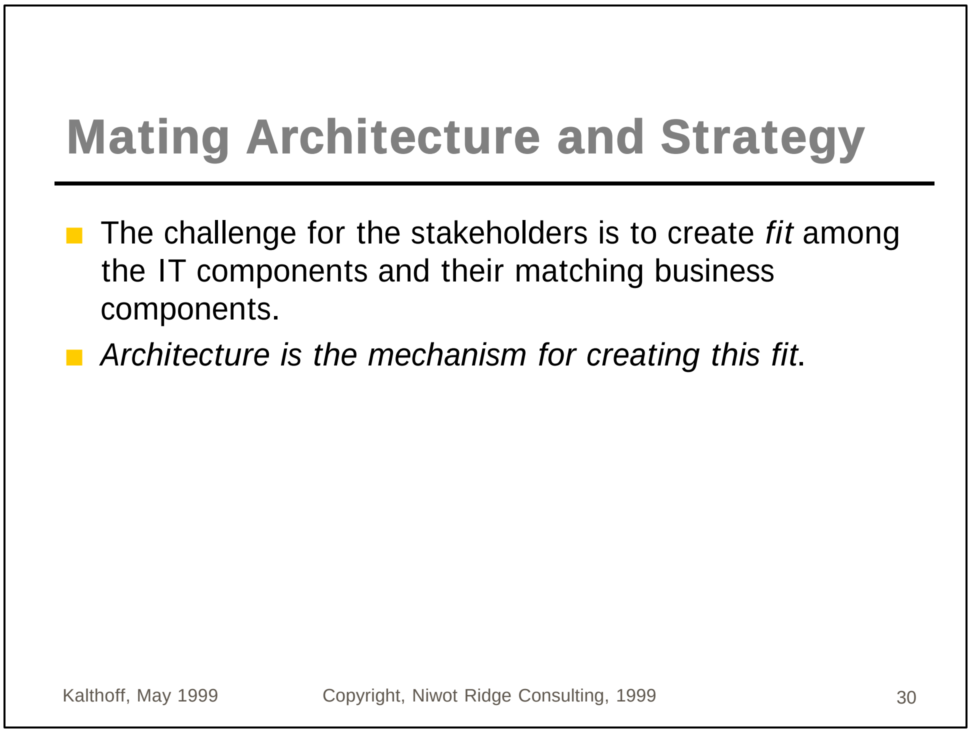


What is Strategy?

- Fit among a company's activities creates pressures and incentives to improve operational effectiveness.
- Fit means that poor performance in one activity will degrade the performance in others, so that weaknesses are exposed drawing management's attention.
- With increasing fit, improvements in one activity will pay dividends in other areas.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 29

- The key to IT strategy is creating fit among the activities of the organization.
- This approach is well developed in the current business literature.
- You may ask yourself, how does my organization think about strategy and fit? Are we confused between strategy and operational effectiveness?

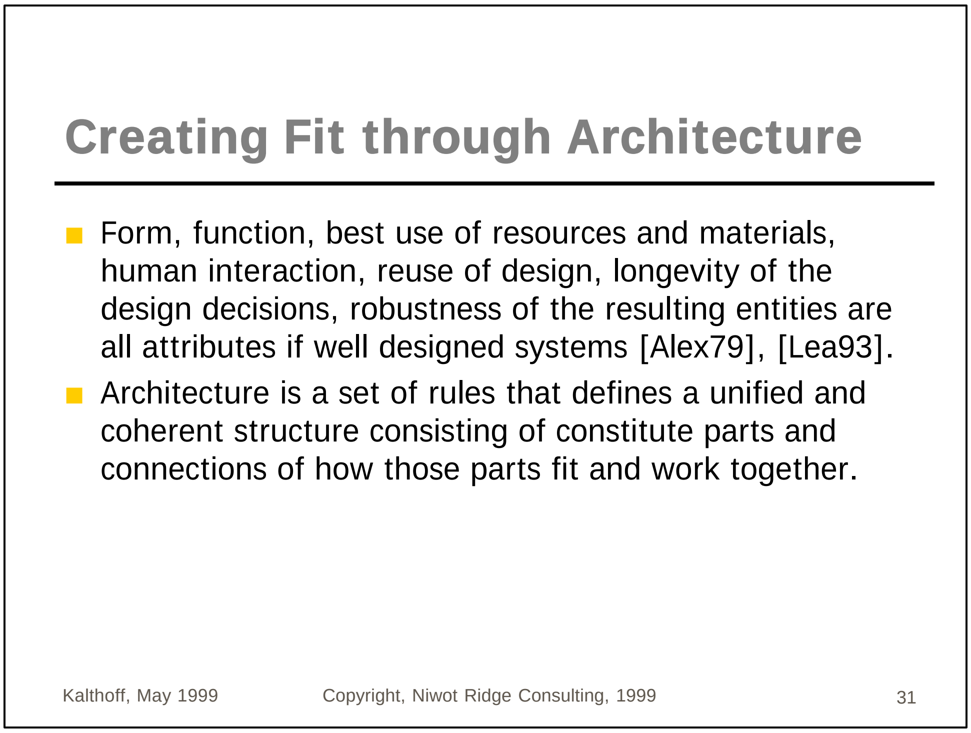


Mating Architecture and Strategy

- The challenge for the stakeholders is to create *fit* among the IT components and their matching business components.
- *Architecture is the mechanism for creating this fit*.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 30

- Now the problem should be clearer…how does architecture help strategy?
- Without strategy and the architecture which which enables it, operational effectiveness is the only outcome.
- We can do things better and faster, so everyone get back to work.
- But what about the future. What about changes in the requirements? What about changes in the market place? What about Internet time?

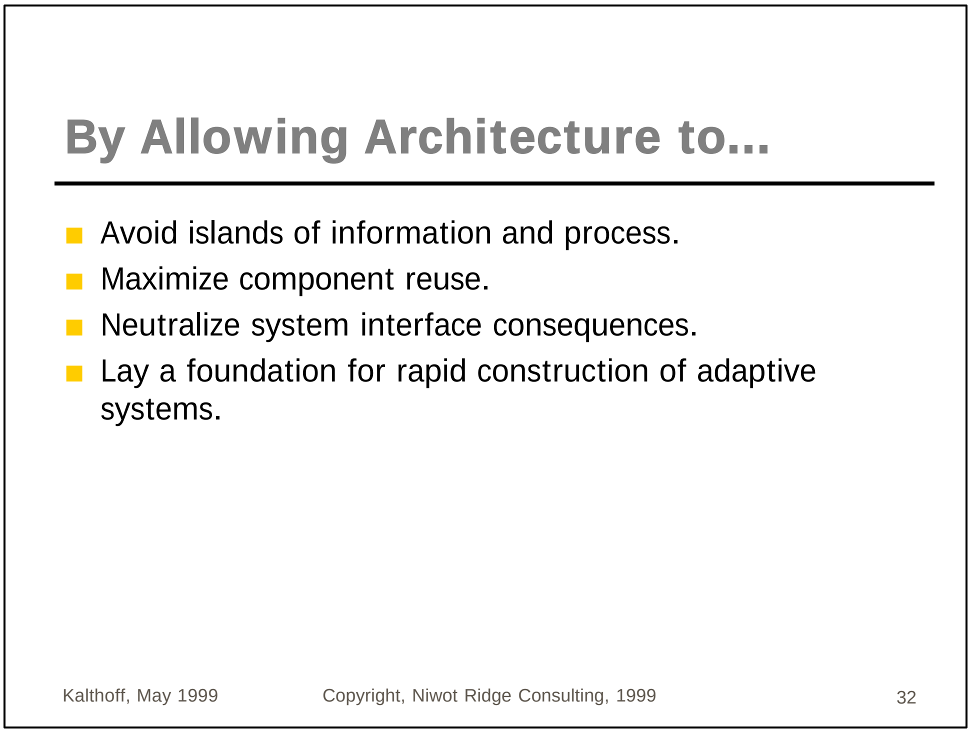


Creating Fit through Architecture

- Form, function, best use of resources and materials, human interaction, reuse of design, longevity of the design decisions, robustness of the resulting entities are all attributes if well designed systems [Alex79], [Lea93].
- Architecture is a set of rules that defines a unified and coherent structure consisting of constitute parts and connections of how those parts fit and work together.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 31

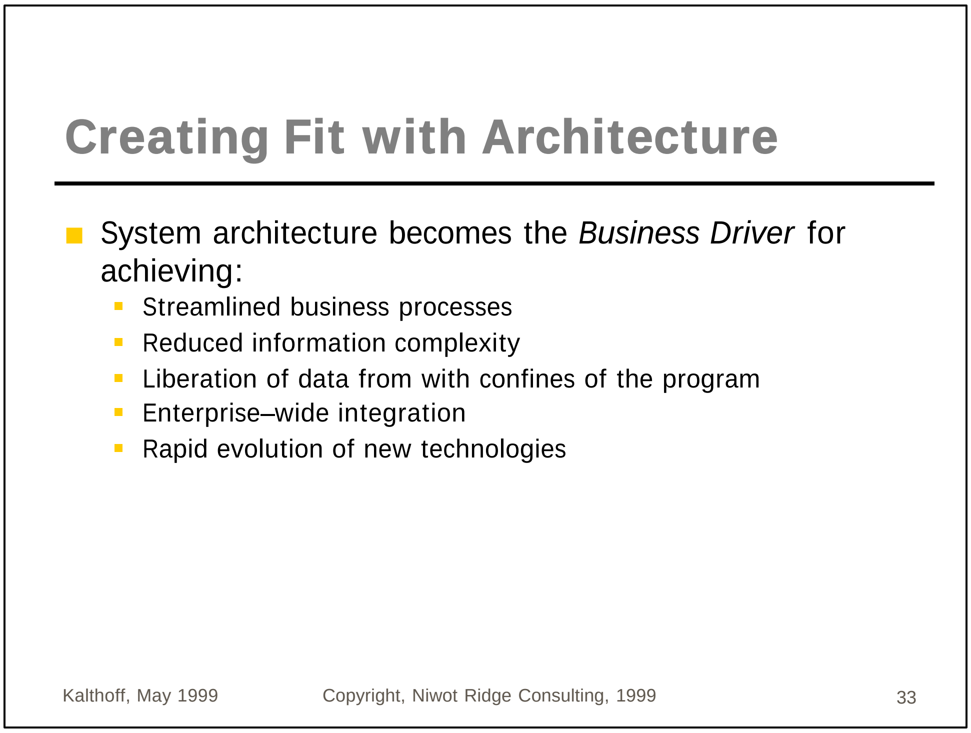
- So here's how to put architecture to work creating fit.
- I would recommend you read something from Alexander. There is a reference in the back of the presentation from Doug Lea of SUNY. This is a good place to start.



By Allowing Architecture to...

- Avoid islands of information and process.
- Maximize component reuse.
- Neutralize system interface consequences.
- Lay a foundation for rapid construction of adaptive systems.

- Here’s the consequences of mating strategy and architecture.

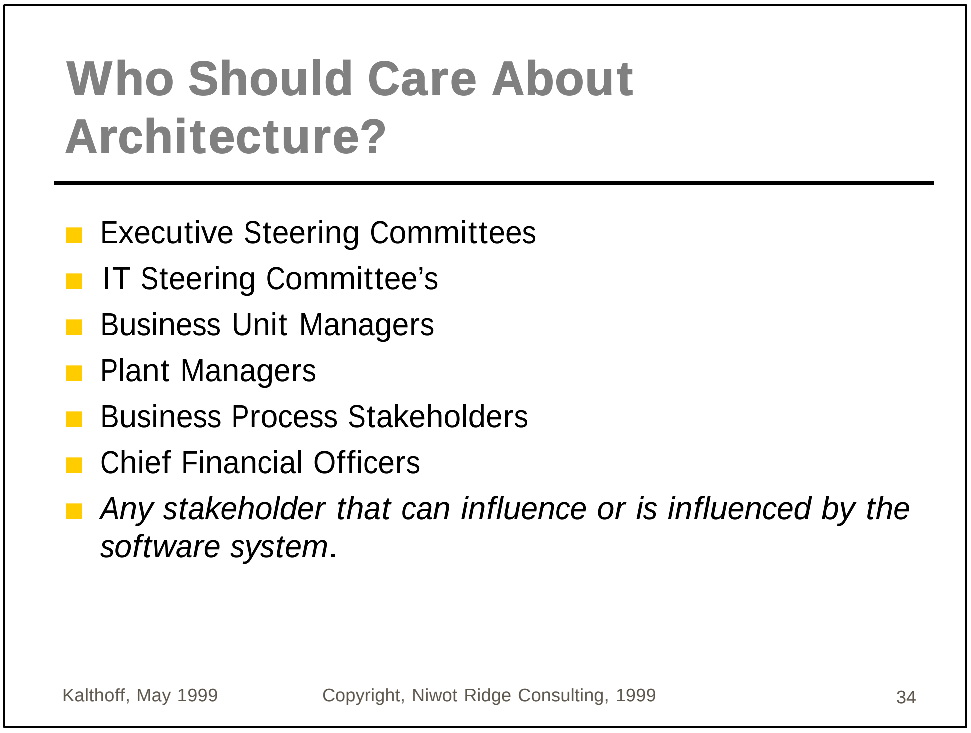


Creating Fit with Architecture

- System architecture becomes the *Business Driver* for achieving:
  - Streamlined business processes
  - Reduced information complexity
  - Liberation of data from with confines of the program
  - Enterprise–wide integration
  - Rapid evolution of new technologies

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 33

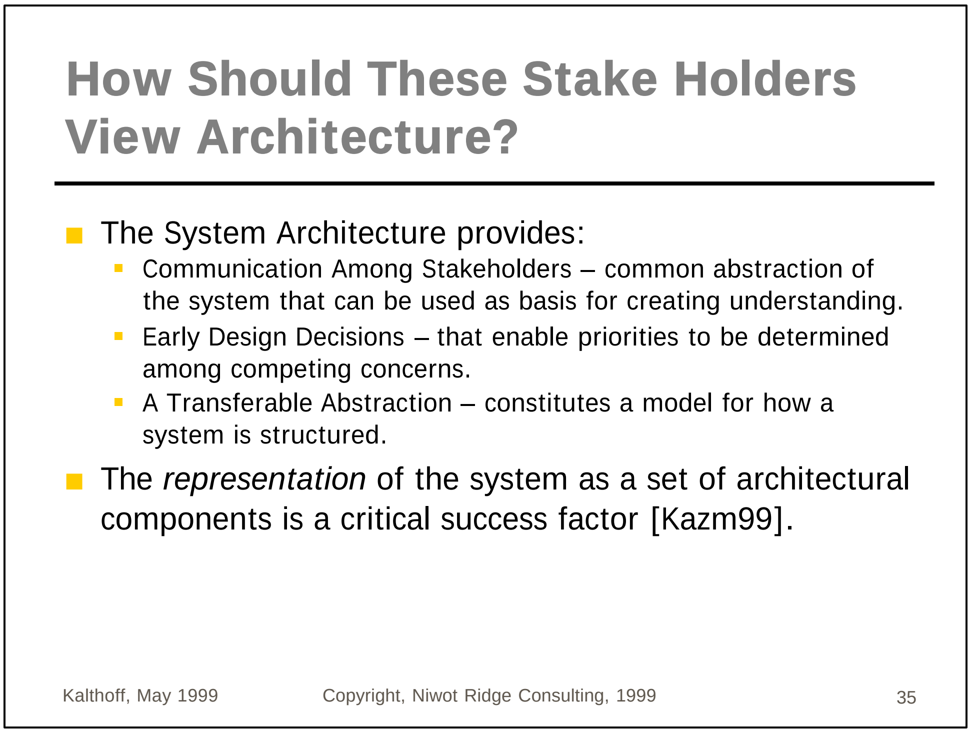
- If all this ethereal talk of building architecture conveying feelings and building towns that express the culture of the residents has you ready to leave, let's switch to balance sheet words.
- Architecture influences business activities in the following way.
- Architecture leverages MONEY, the ultimate motivator of our society. This is not a political statement, just a clarification of our goals.


Who Should Care About Architecture?

- Executive Steering Committees
- IT Steering Committee's
- Business Unit Managers
- Plant Managers
- Business Process Stakeholders
- Chief Financial Officers
- *Any stakeholder that can influence or is influenced by the software system*.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 34

- Now that we've got the proper motivation on the table, who should be involved in this very practical endeavor?
- Here's a starting list. These folks are motivated by money. I don't necessarily mean personal money (but bonuses may be involved), I mean money in terms of business profits, earnings, expense reduction, return on capital ... all that kind of money.

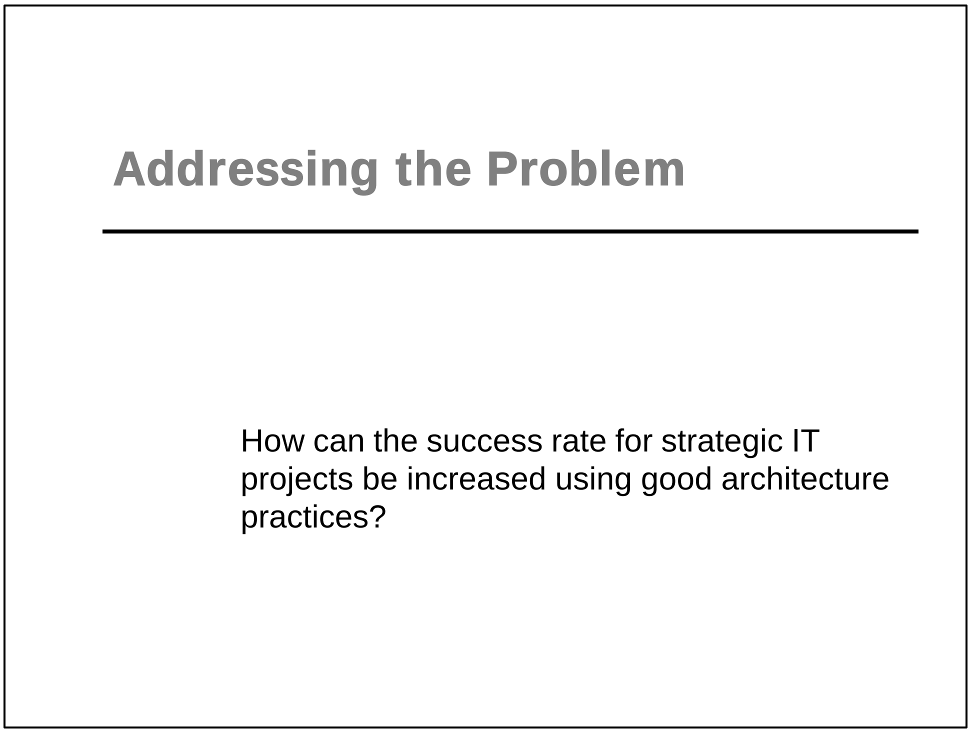


How Should These Stake Holders View Architecture?

- The System Architecture provides:
  - Communication Among Stakeholders – common abstraction of the system that can be used as basis for creating understanding.
  - Early Design Decisions – that enable priorities to be determined among competing concerns.
  - A Transferable Abstraction – constitutes a model for how a system is structured.
- The *representation* of the system as a set of architectural components is a critical success factor [Kazm99].

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 35

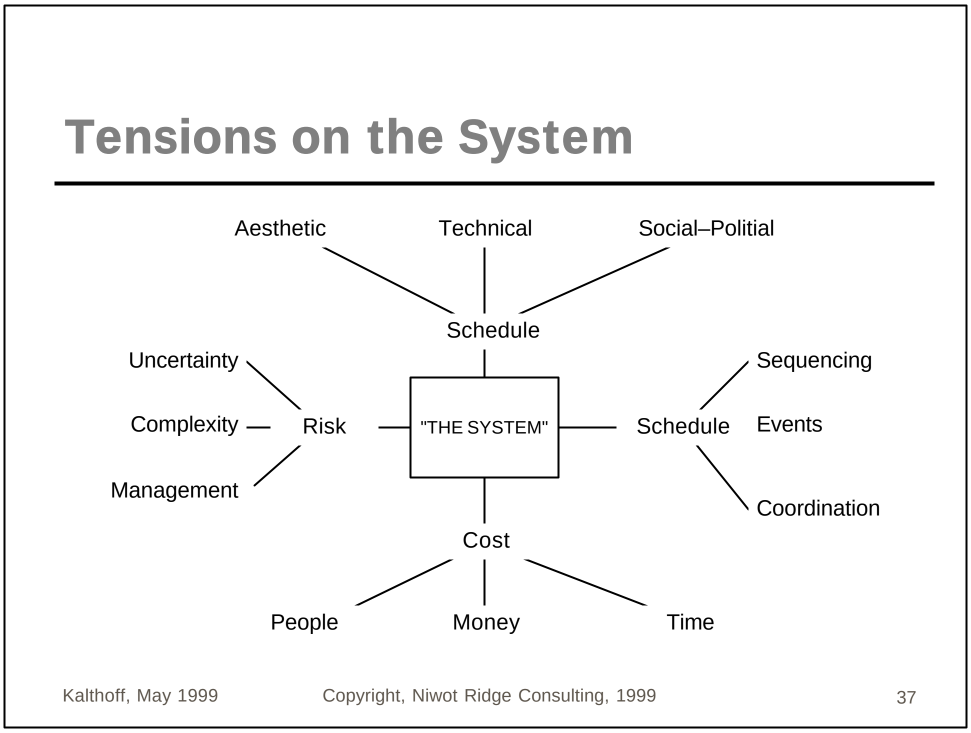
- Now that we have the stakeholders identified, how can architecture affect their lives?
- The architecture of a system can be used to ....
- A representation of the system can be constructed and be used to evaluate the consequences of business decision, without necessarily having to build an actual system.
- The stakeholder can *play around* with the system ... in some ways



Addressing the Problem

How can the success rate for strategic IT projects be increased using good architecture practices?

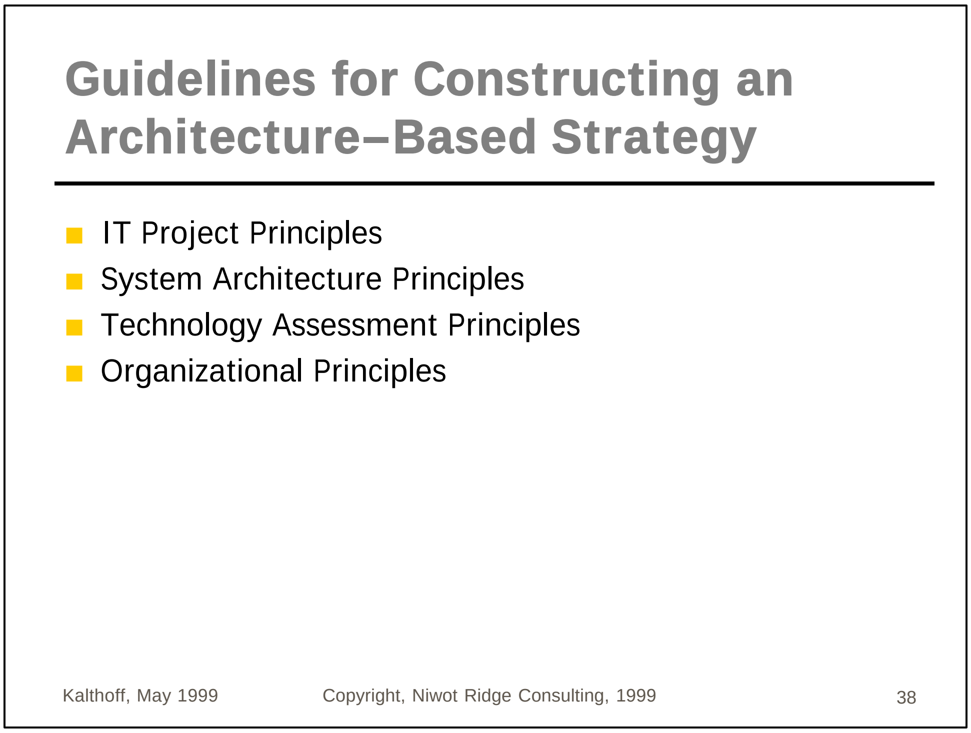
- Ok, now that we have some idea about the words and the setting, what are the opportunities for actually doing something worth while?
- We're all here to hopefully take back answers, not just hear a bunch of words.



Tensions on the System

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 37

- Like the forces on the Project Manager, these are forces on the project itself.
- Many of these forces deal with politics of IT as well as the technology.

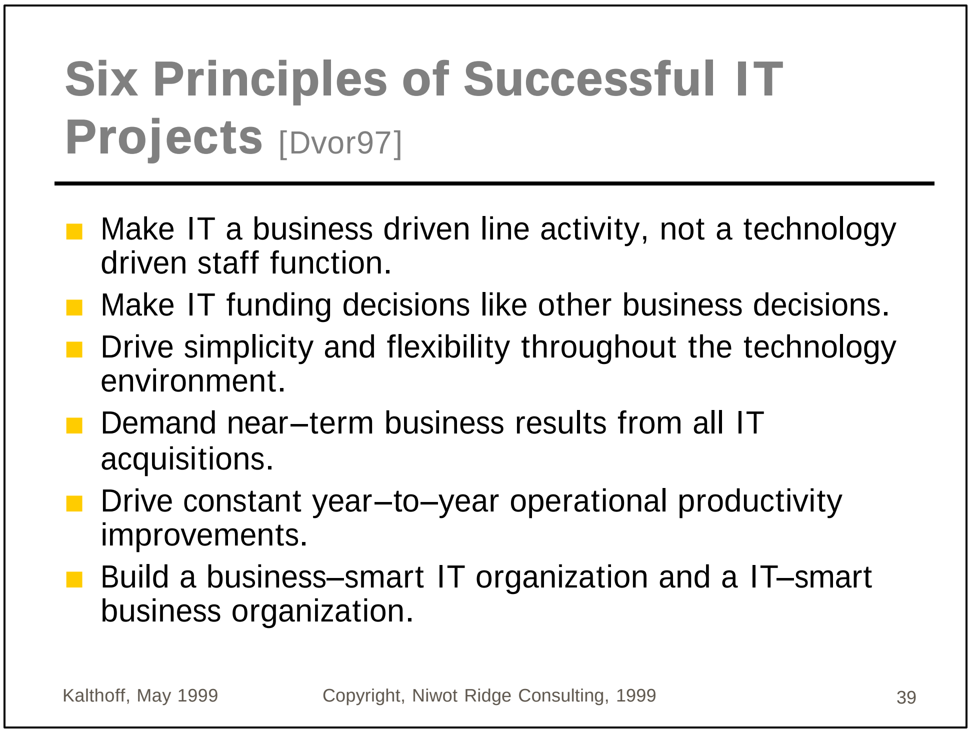


Guidelines for Constructing an Architecture–Based Strategy

- IT Project Principles
- System Architecture Principles
- Technology Assessment Principles
- Organizational Principles

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 38

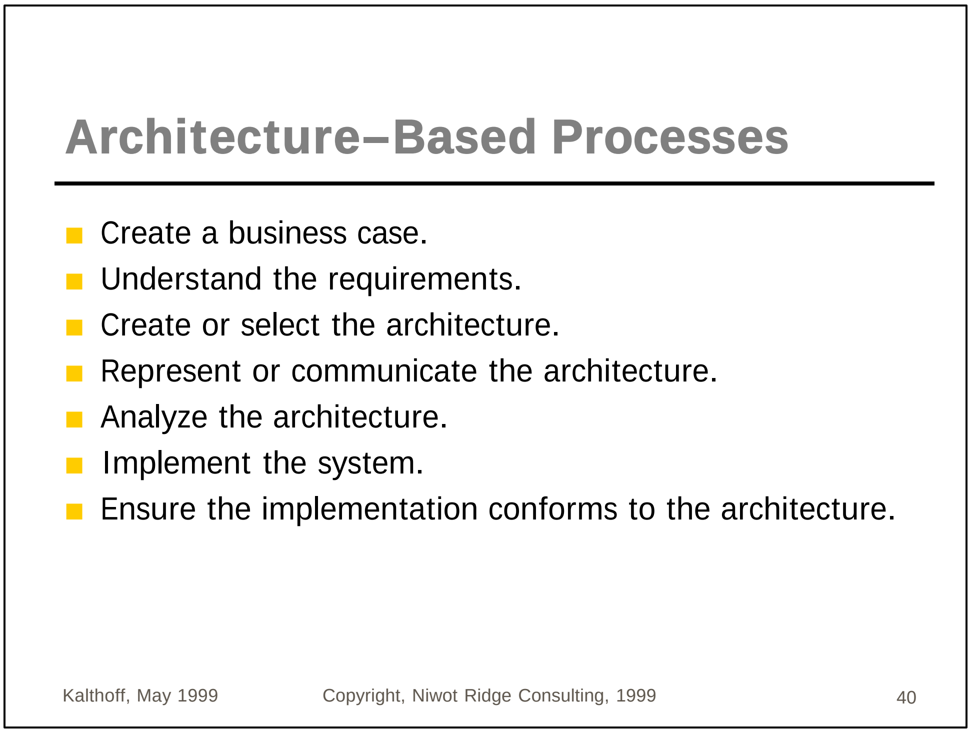
- There are several principles that can be used to guide us through the architecture process.
- Some are focused on the technical aspects of the system and some on the business and strategy process.
- Each of these principles is just a set of guidelines. You can make up your own, copy these or borrow from other sources.
- No matter what the source, the principles should be shared by all the participants and kept in view during the duration of the projects.


Six Principles of Successful IT Projects [Dvor97]

- Make IT a business driven line activity, not a technology driven staff function.
- Make IT funding decisions like other business decisions.
- Drive simplicity and flexibility throughout the technology environment.
- Demand near–term business results from all IT acquisitions.
- Drive constant year–to–year operational productivity improvements.
- Build a business–smart IT organization and a IT–smart business organization.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 39

- Here are some IT Project principles.
- They are obvious now that they're written down, but maybe not so obvious before they appeared on paper.
- If they were obvious AND of projects lived by them, then we would not be here today, would we?
- So one other principle should be that *what appears easy at first may actually turn out to be hard in the end*.

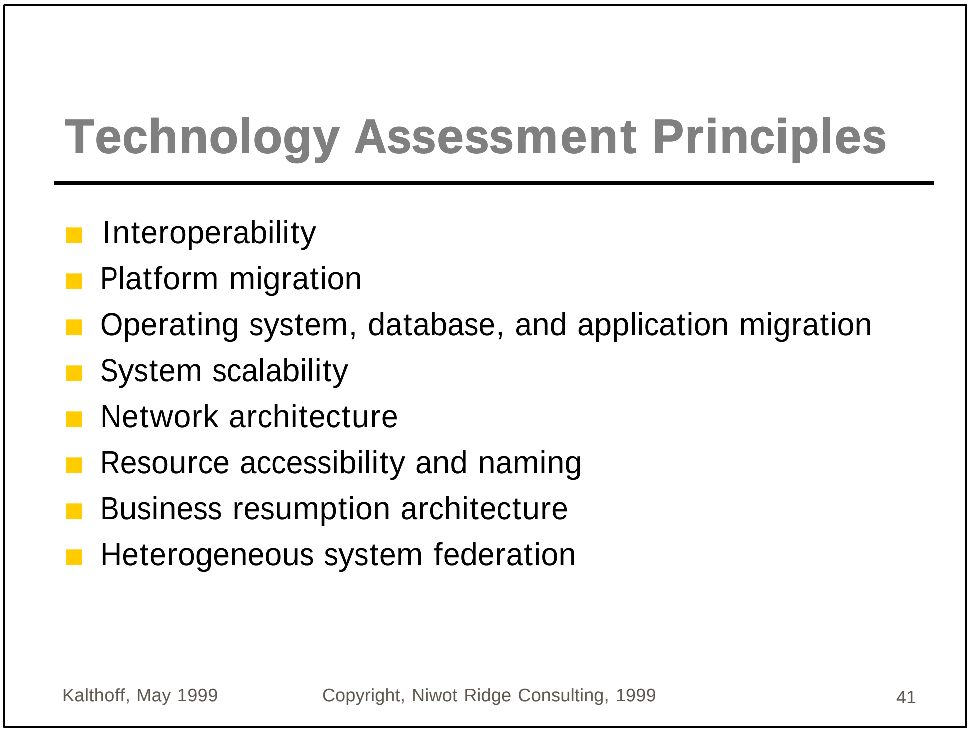


Architecture–Based Processes

- Create a business case.
- Understand the requirements.
- Create or select the architecture.
- Represent or communicate the architecture.
- Analyze the architecture.
- Implement the system.
- Ensure the implementation conforms to the architecture.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 40

- Create business is broader than assessing the need for the system. It constrains the requirements.
- There are many ways to understand the requirements. Use cases is one that end user like.
- The conceptual integrity is key to sound system architecture. This integrity can only be held by a small number of minds.
- For architecture to be effective it must be communicated.
- Some form of analysis must be performed. Scenario based methods are powerful.
- Keeping the developers and procurement process faithful to the architecure is the primary role of the ARB, architect and the project manager.

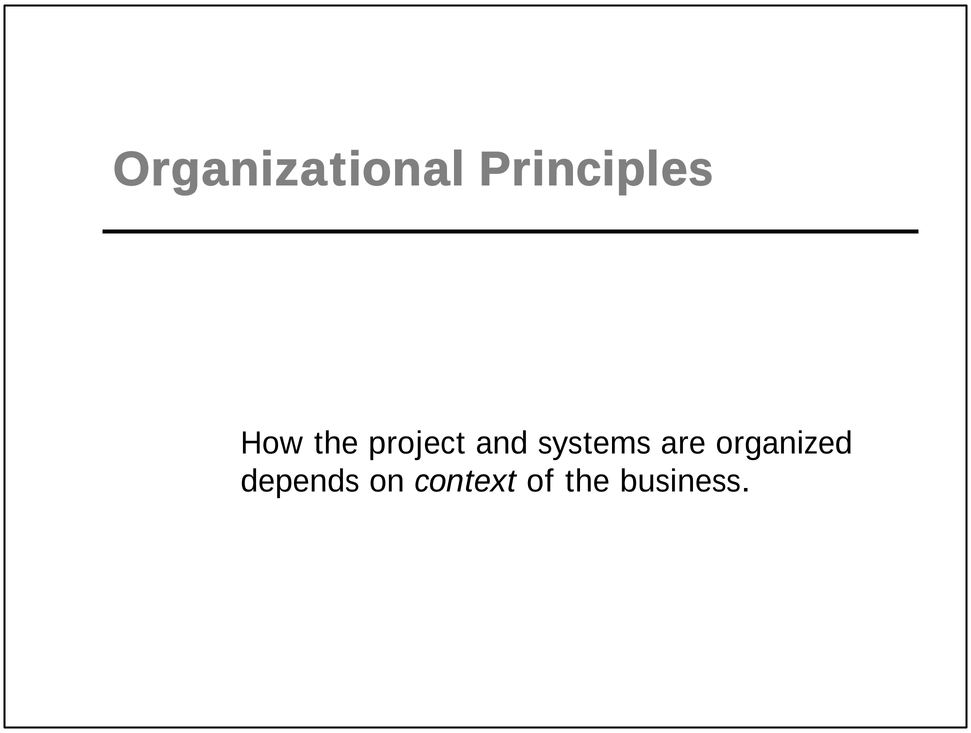


Technology Assessment Principles

- Interoperability
- Platform migration
- Operating system, database, and application migration
- System scalability
- Network architecture
- Resource accessibility and naming
- Business resumption architecture
- Heterogeneous system federation

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 41

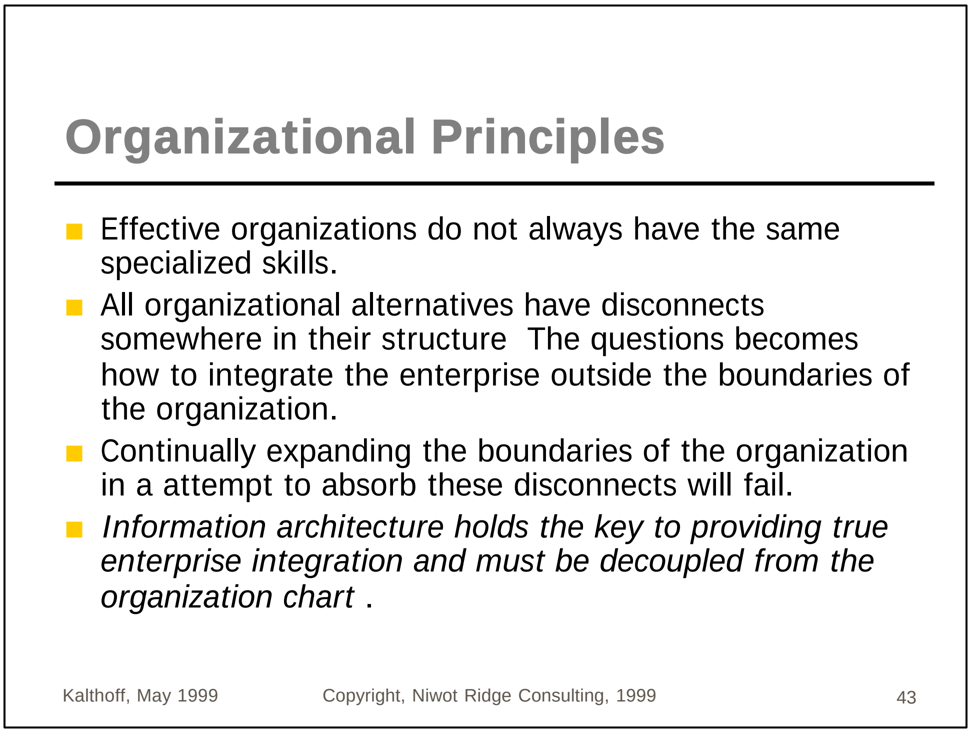
- The principles used in assessing technology architecture are pretty well defined by the technology vendors and the academics that work in this area.
- Here are some areas where questions need to be asked. Especially in the areas where migration from one platform environment to the other may take place.
- There is a big question when Business Continuation is considered.
  - How will the system actually work at some other site?
  - Can the business continuation process be tested?
  - Is there a plan B if the first attempts fail?



Organizational Principles

How the project and systems are organized depends on *context* of the business.

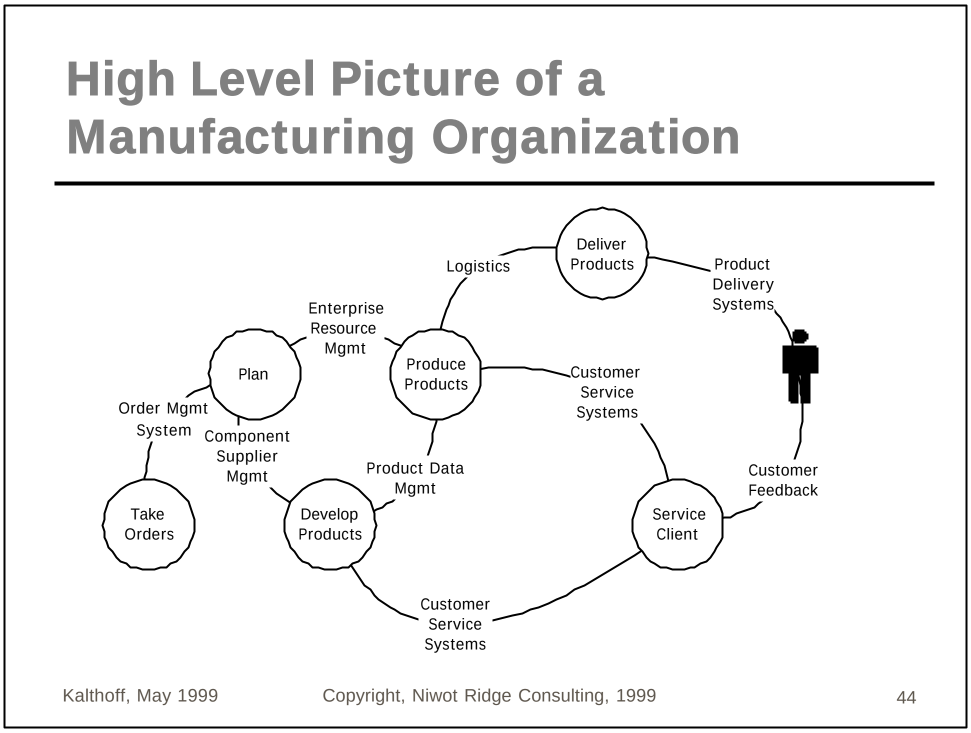
- Many of the technology and architectural issues can be traced to the business organization.
- We all have experiences with our own organizations (current or past) and their dysfunctional members and methods


Organizational Principles

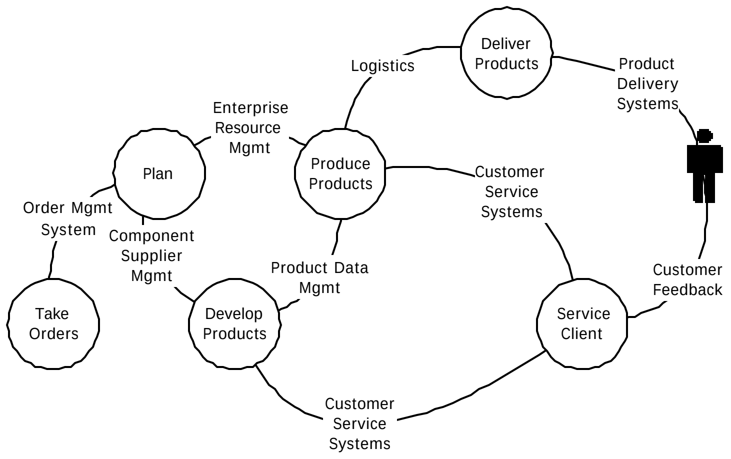
- Effective organizations do not always have the same specialized skills.
- All organizational alternatives have disconnects somewhere in their structure The questions becomes how to integrate the enterprise outside the boundaries of the organization.
- Continually expanding the boundaries of the organization in a attempt to absorb these disconnects will fail.
- *Information architecture holds the key to providing true enterprise integration and must be decoupled from the organization chart* .

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 43

- Organization is an area that has a rich field of research.
- Here are some principles that can be applied to IT projects.
- We’re back to having the architecture of the system influence the organization structure.
- This is a principle that is overlooked in the majority of projects.
  - The people doing the work should be organized in some way to match the system they are building.
  - This is vital to the success of the project, if the developers are not in some way connected with the customers.

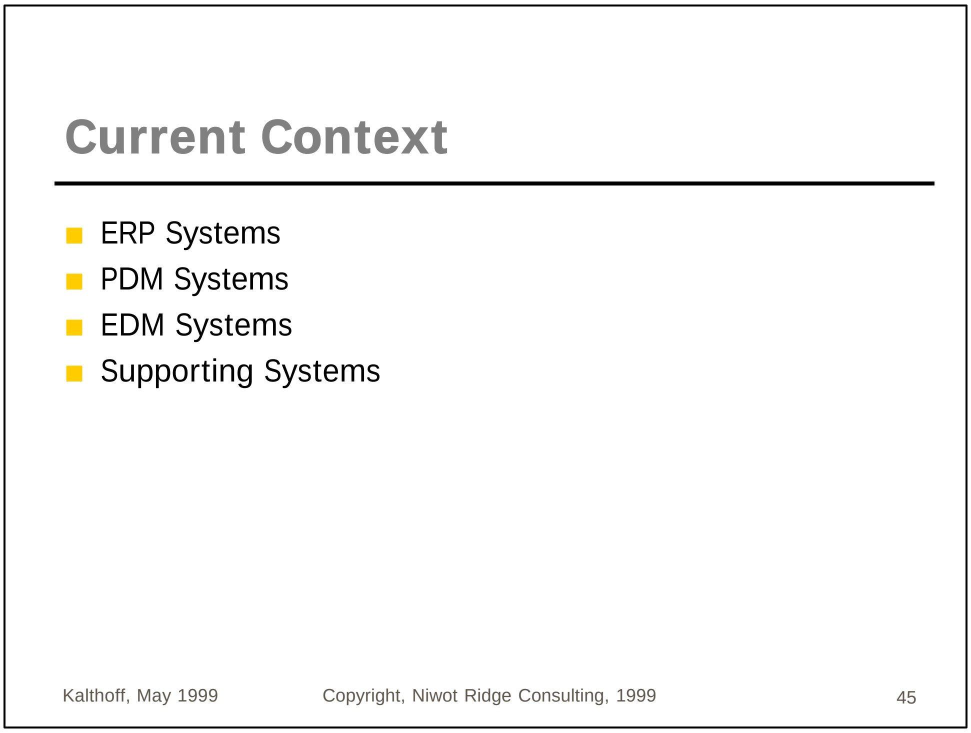


High Level Picture of a Manufacturing Organization



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 44

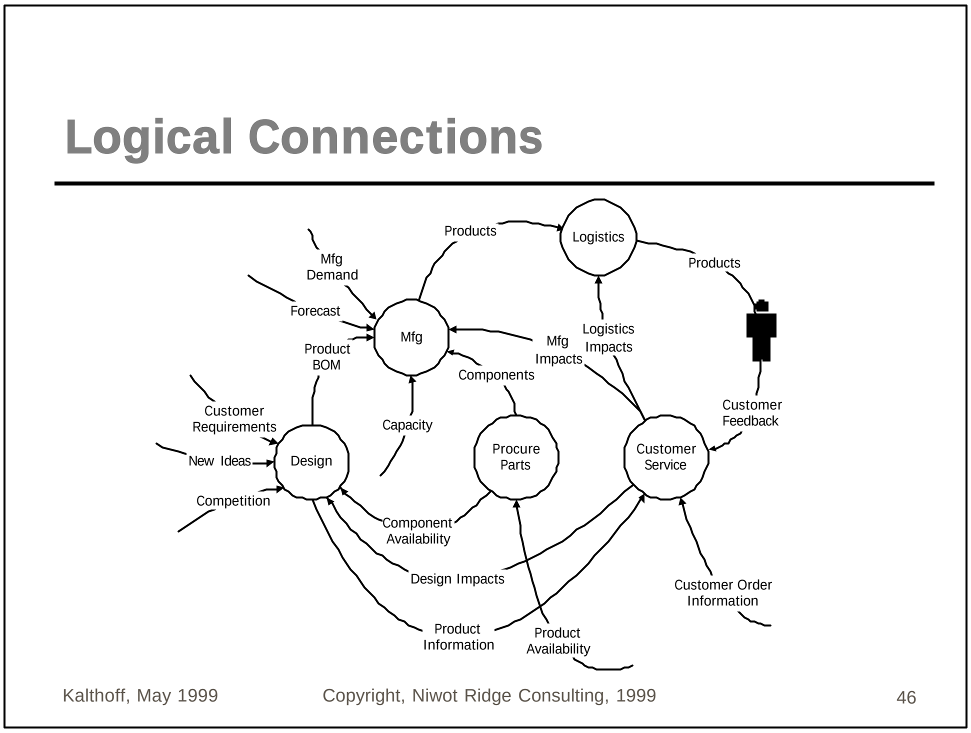
- Here's a very simple model of a manufacturing organization with diverse business organizations, various forces and tensions and varying requirements.



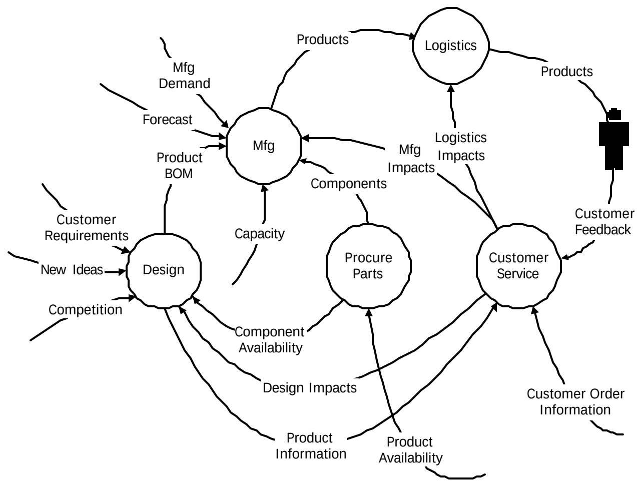
Current Context

- ERP Systems
- PDM Systems
- EDM Systems
- Supporting Systems

- This business organization can have many systems laid on top of it.
- These include ERP, PDM, EDM, and their supporting or glue systems.

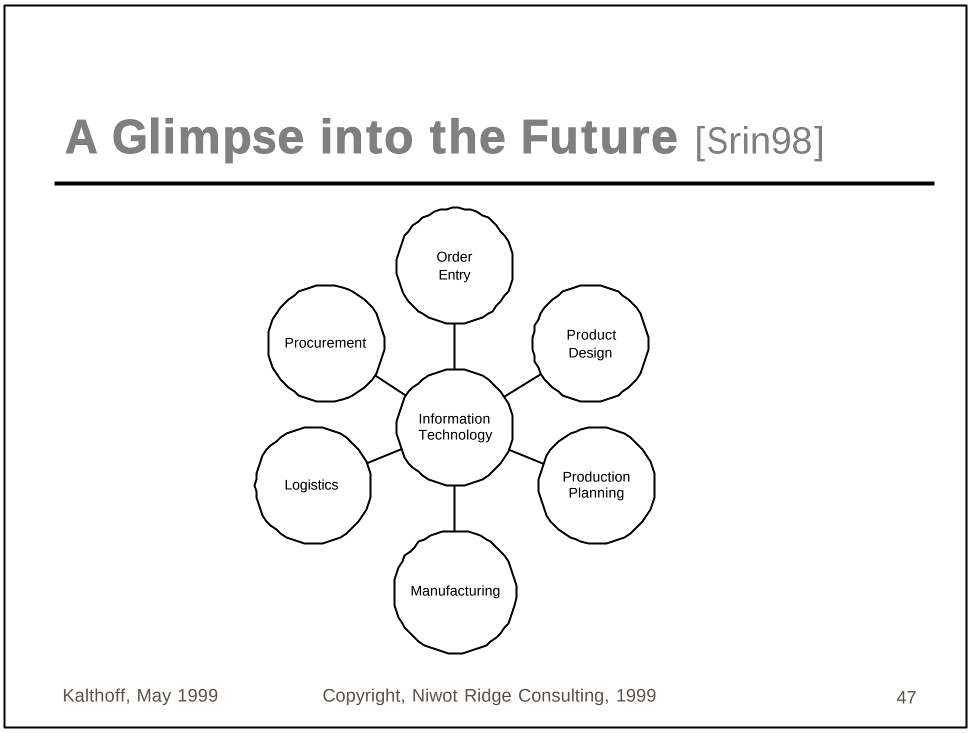


Logical Connections

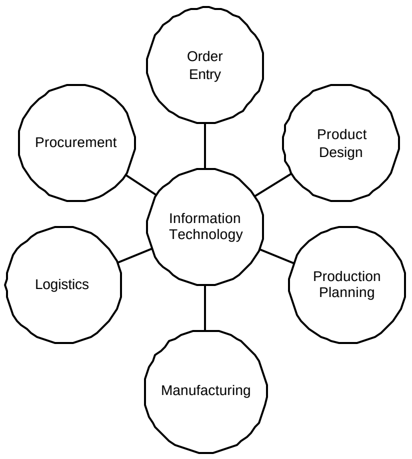


Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 46

- Now we can start to see these dependencies across the organizational boundaries.
- Each is these *infrastructure* systems interacts with various business functions and their various influences.
- From the VERY simple set of diagrams the level of complexity grows rapidly.
- Some times is grows so rapidly it simply flies apart.

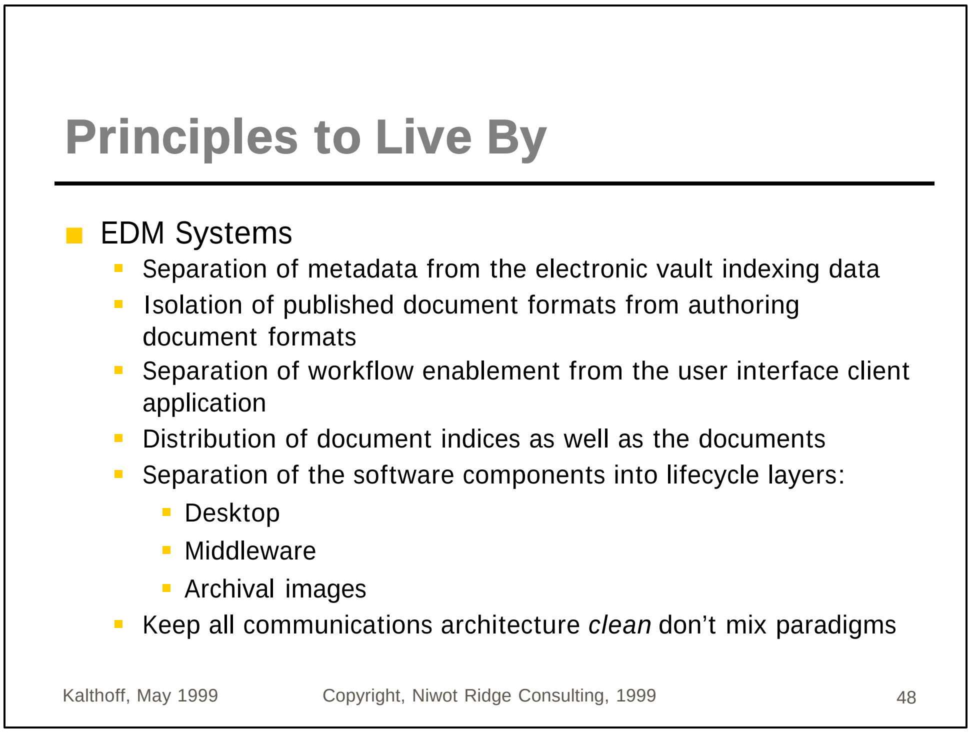


A Glimpse into the Future [Srin98]



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 47

- In the ideal world all the components of the manufacturing environment would interoperate in a seamless manner.
- In reality the seams are made from fairly large stitches.
- The results sometimes look like Frankenstein.
- But the approach is fundamentally correct, in the CORBA world this is the foundation of the seamless environment.
- The SemaTech architecture will show how this can be done.

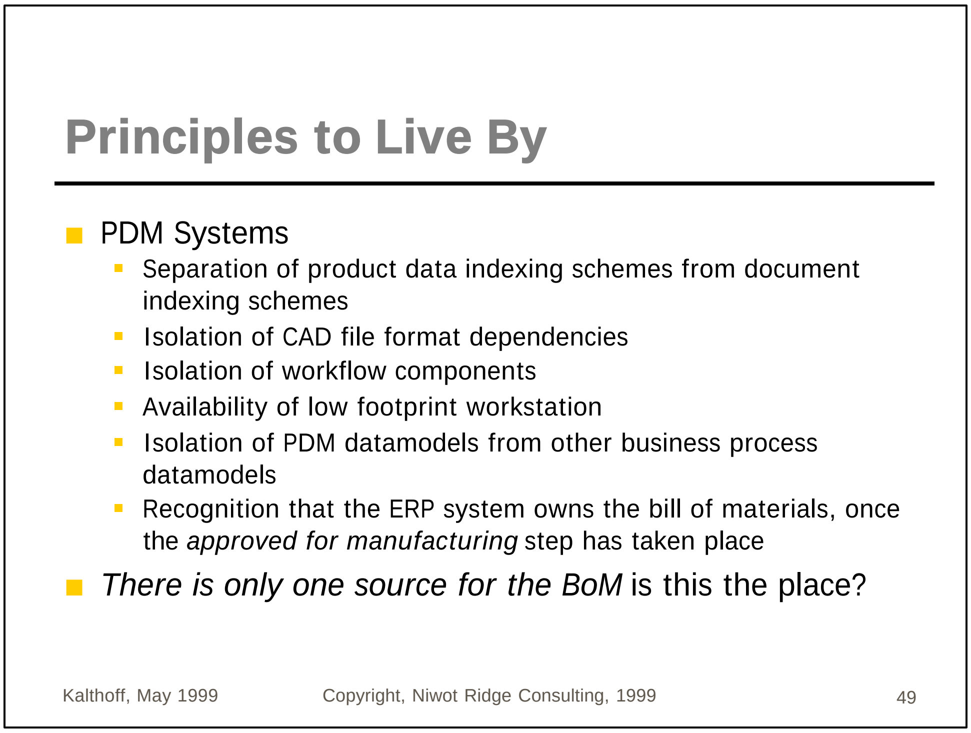


Principles to Live By

- EDM Systems
  - Separation of metadata from the electronic vault indexing data
  - Isolation of published document formats from authoring document formats
  - Separation of workflow enablement from the user interface client application
  - Distribution of document indices as well as the documents
  - Separation of the software components into lifecycle layers:
    - Desktop
    - Middleware
    - Archival images
  - Keep all communications architecture *clean* don't mix paradigms

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 48

- Now we come to some principles for each solution domain. This is a very small space to list these principles, but it is a start.
- The EDM System principles are VERY HIGH LEVEL, so this is just the start.
- The last one is not so obvious. In the world of the Web, mixing the paradigms can have disastrous results. For example
  - Having a pure Web architecture with JDBC connections
  - Having a Unix strategy with that lonely NT server
  - Creating archival grade documents in a native file format....What was the most popular WP in 1992?

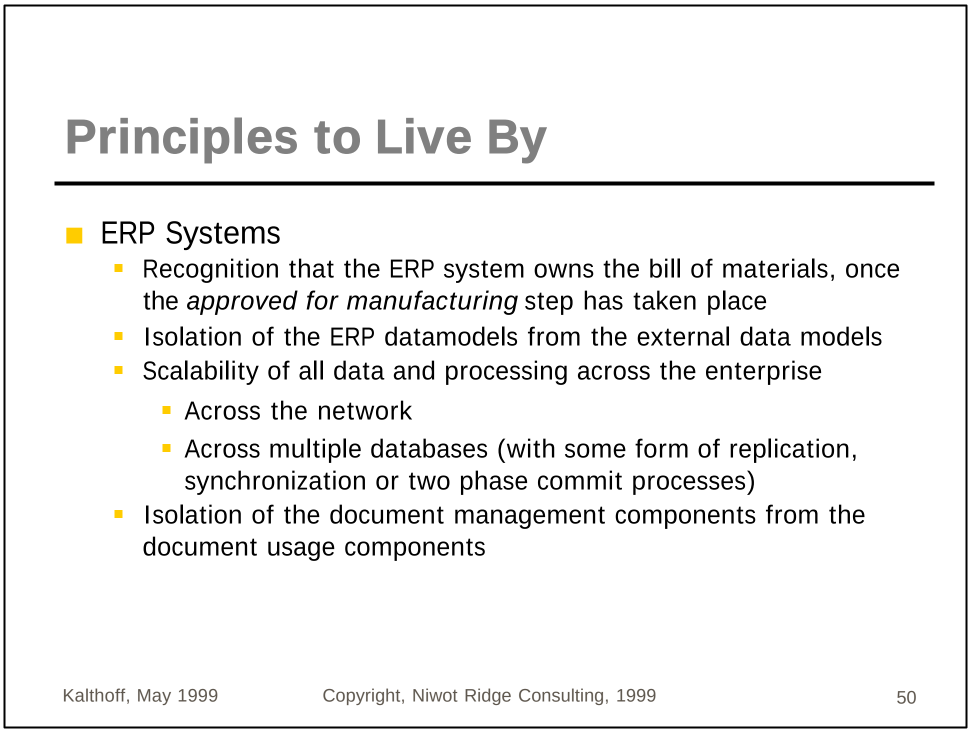


Principles to Live By

- PDM Systems
  - Separation of product data indexing schemes from document indexing schemes
  - Isolation of CAD file format dependencies
  - Isolation of workflow components
  - Availability of low footprint workstation
  - Isolation of PDM datamodels from other business process datamodels
  - Recognition that the ERP system owns the bill of materials, once the *approved for manufacturing* step has taken place
- *There is only one source for the BoM* is this the place?

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 49

- The PDM world is growing more complex by the day. The ERP systems are targeting the traditional PDM market place, since they've already run over all the stand alone applications they can in the MRP, order entry, domain.
- The key principle here is to understand what PDM systems do best and let them do it.
  - This is usually not EDM or ERP like functions
  - PDM systems are product structure centric.
  - If your problem is not product structure centric, the PDM architecture is the wrong fit.
- The *killer* principle is to determine where the Bill of Materials will be authored, maintained, updated, etc. There is no obvious answer, no matter what the sales guys tell you.

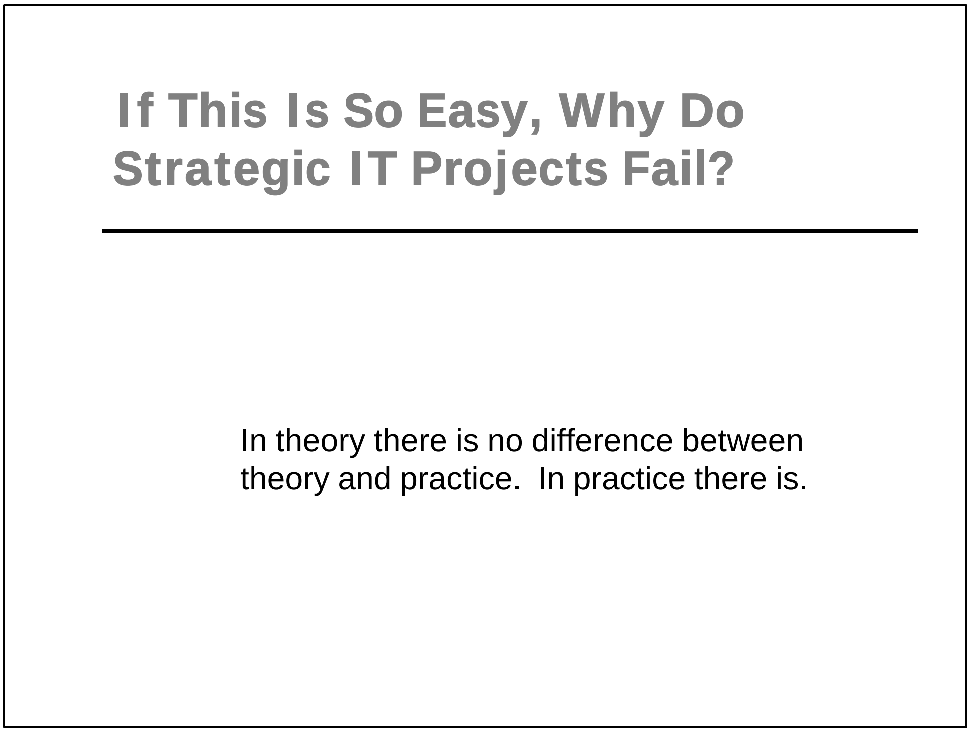


Principles to Live By

- ERP Systems
  - Recognition that the ERP system owns the bill of materials, once the *approved for manufacturing* step has taken place
  - Isolation of the ERP datamodels from the external data models
  - Scalability of all data and processing across the enterprise
    - Across the network
    - Across multiple databases (with some form of replication, synchronization or two phase commit processes)
  - Isolation of the document management components from the document usage components

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 50

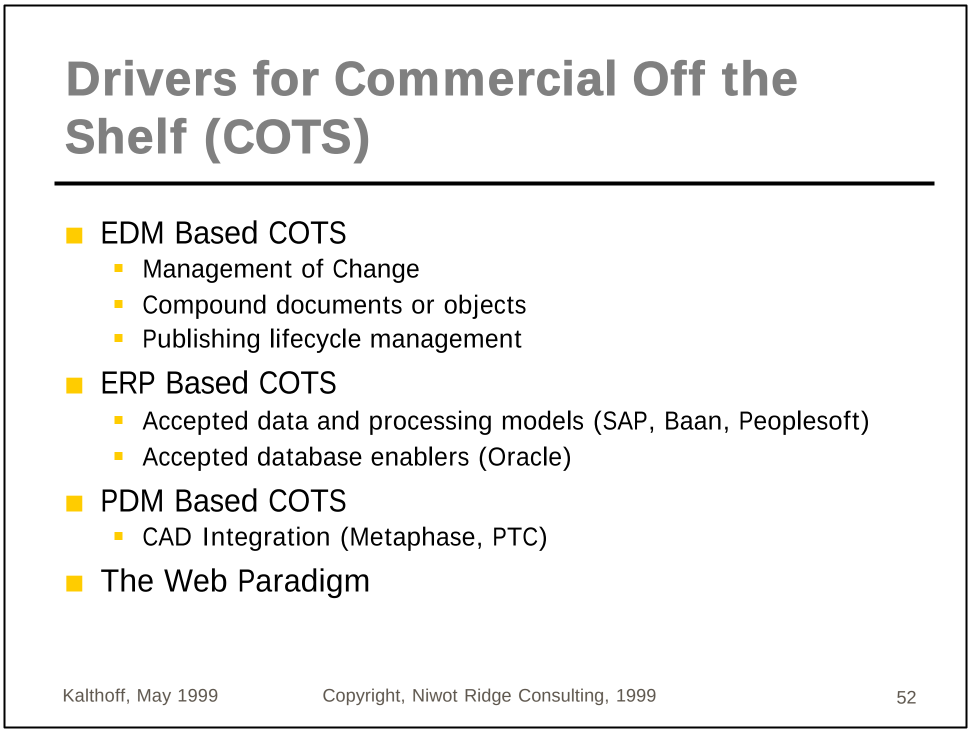
- This is where is gets ugly. The ERP systems want to and some times have to *OWN THE WORLD*.
- Separating this need to own everything from the need to provide specialized processing (PDM and EDM) is the $64,000,000 question -- literally.
- The ERP vendor's architectural paradigm is to be the center of the universe. This is a good strategy and a good architecture, since the system are wickedly complex and the whole world has come to depend on the ERP system.
- The problem is not understanding this, and treating the ERP system as a *peer* to the other systems



If This Is So Easy, Why Do Strategic IT Projects Fail?

In theory there is no difference between theory and practice. In practice there is.

- So, why is there all this chaos in the world?
- Well, it turns out that proactive is harder than theory.

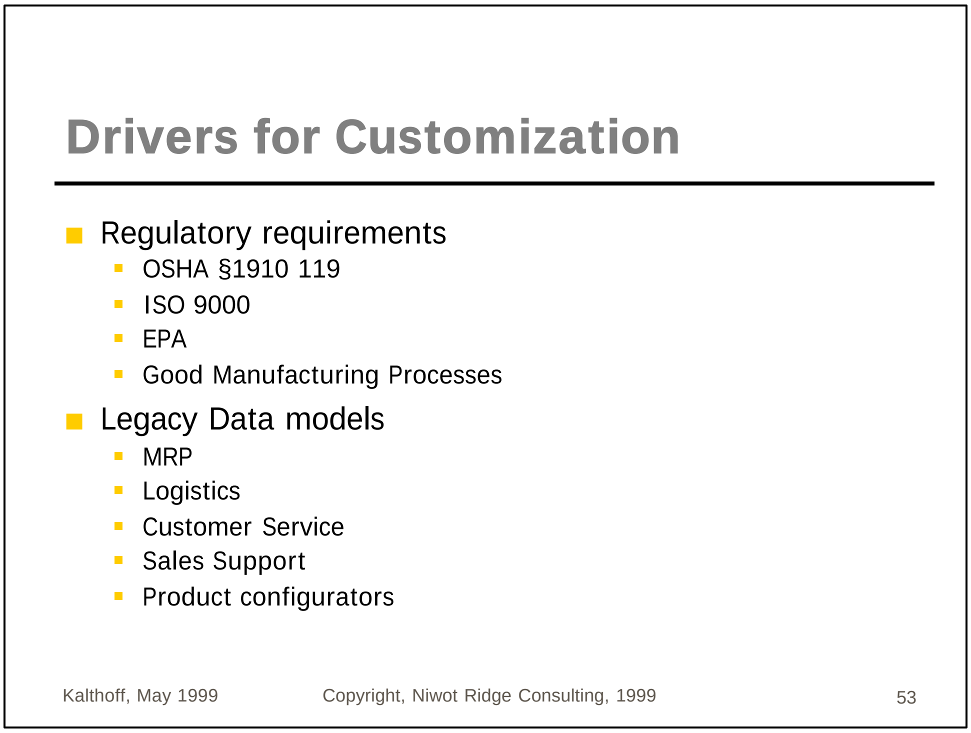


Drivers for Commercial Off the Shelf (COTS)

- EDM Based COTS
  - Management of Change
  - Compound documents or objects
  - Publishing lifecycle management
- ERP Based COTS
  - Accepted data and processing models (SAP, Baan, Peoplesoft)
  - Accepted database enablers (Oracle)
- PDM Based COTS
  - CAD Integration (Metaphase, PTC)
- The Web Paradigm

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 52

- The MoC is the *killer* application for EDM and PDM. The COTS systems provide this out of the box, but there are subtle issues
  - versions and rules for versions
  - revision management
  - where used and used by for versions and revisions
  - associations and ownership of the relationships
  - multiple renditions of the documents and parts descriptions

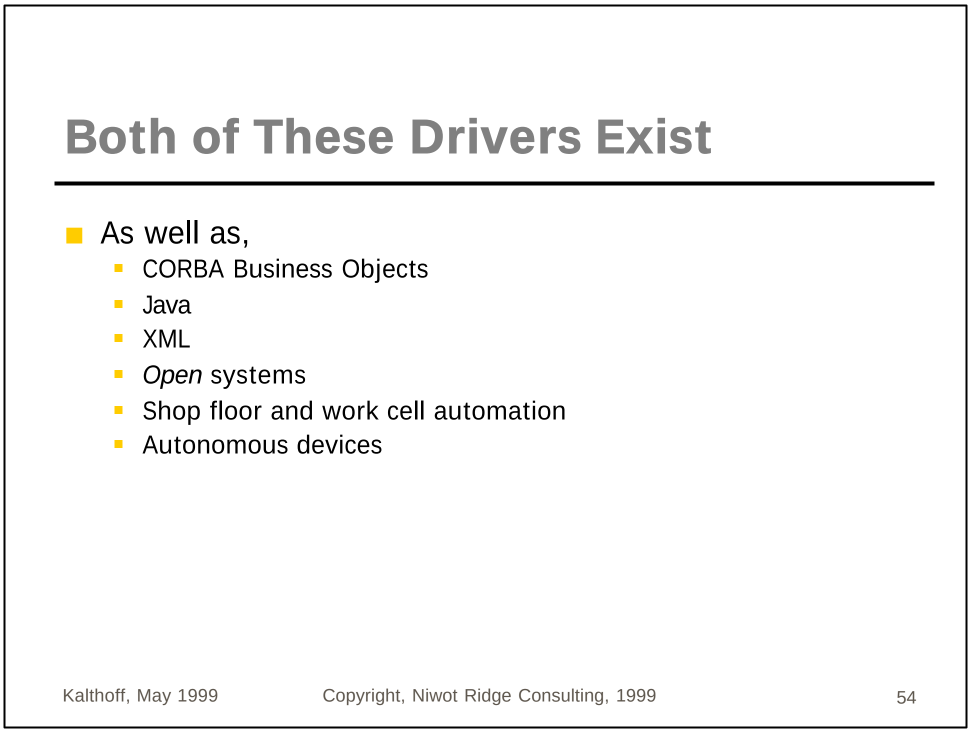


Drivers for Customization

- Regulatory requirements
  - OSHA §1910 119
  - ISO 9000
  - EPA
  - Good Manufacturing Processes
- Legacy Data models
  - MRP
  - Logistics
  - Customer Service
  - Sales Support
  - Product configurators

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 53

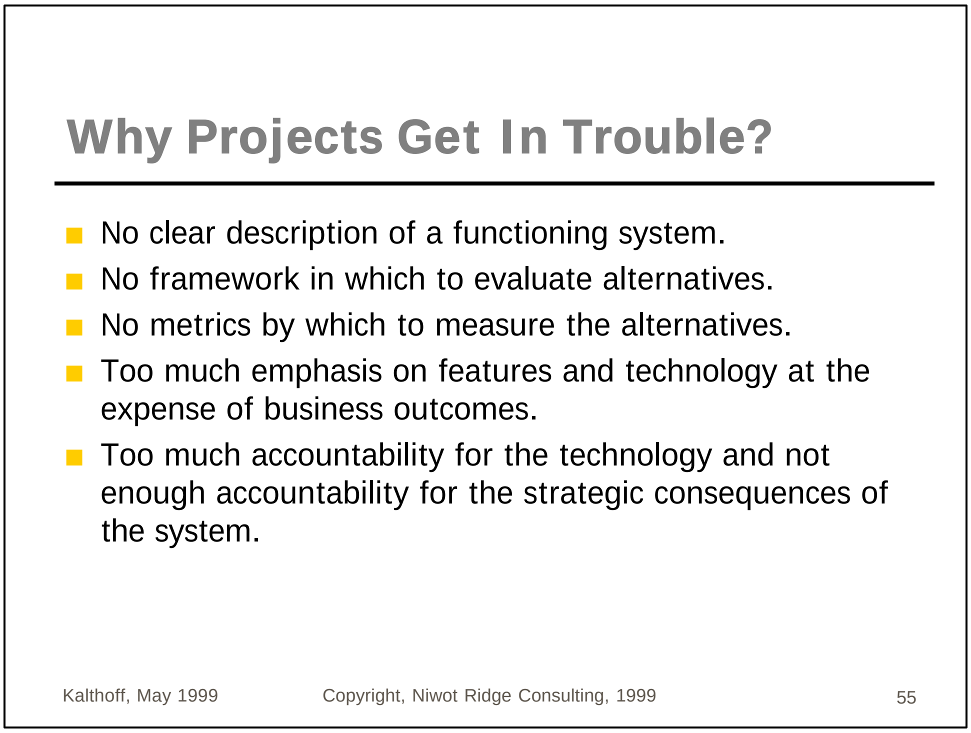
- In many instances there are clear needs for customizing the system to meet specific needs.
- These are usually found in regulatory environments or legacy system integration environments.
- The effort necessary to integrate two systems grows by a substantial factor when other systems are added.
- The architecture of each of these integrated systems must somehow be made compatible with all the other systems.


Both of These Drivers Exist

- As well as,
  - CORBA Business Objects
  - Java
  - XML
  - *Open* systems
  - Shop floor and work cell automation
  - Autonomous devices

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 54

- Actually both drivers exist in the real world, since the legacy systems and the COTS features are needed.
- Here are some more drivers which will complicate the issue.

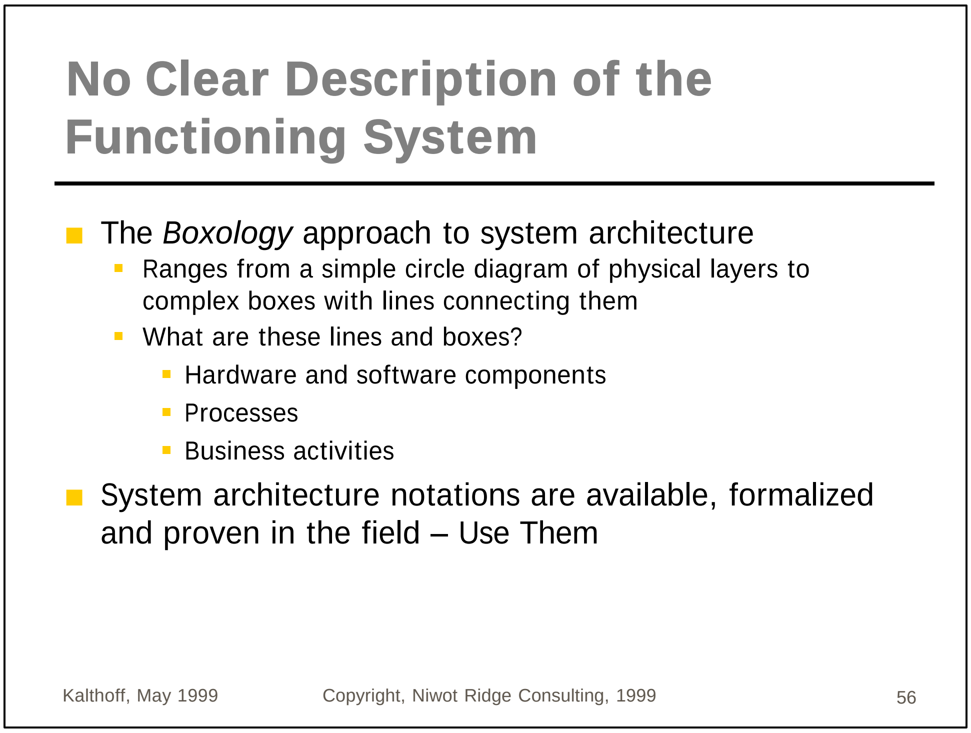


Why Projects Get In Trouble?

- No clear description of a functioning system.
- No framework in which to evaluate alternatives.
- No metrics by which to measure the alternatives.
- Too much emphasis on features and technology at the expense of business outcomes.
- Too much accountability for the technology and not enough accountability for the strategic consequences of the system.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 55

- There are more examples of project failure than there are project success.
- The research shows the following items are the primary causes of project failures.
- There are many sources of this research, including the Standish Group, which tends to be pessimistic, the SEI which tends to be optimistic.
- No matter what their views the numbers can be used to derive these factors.

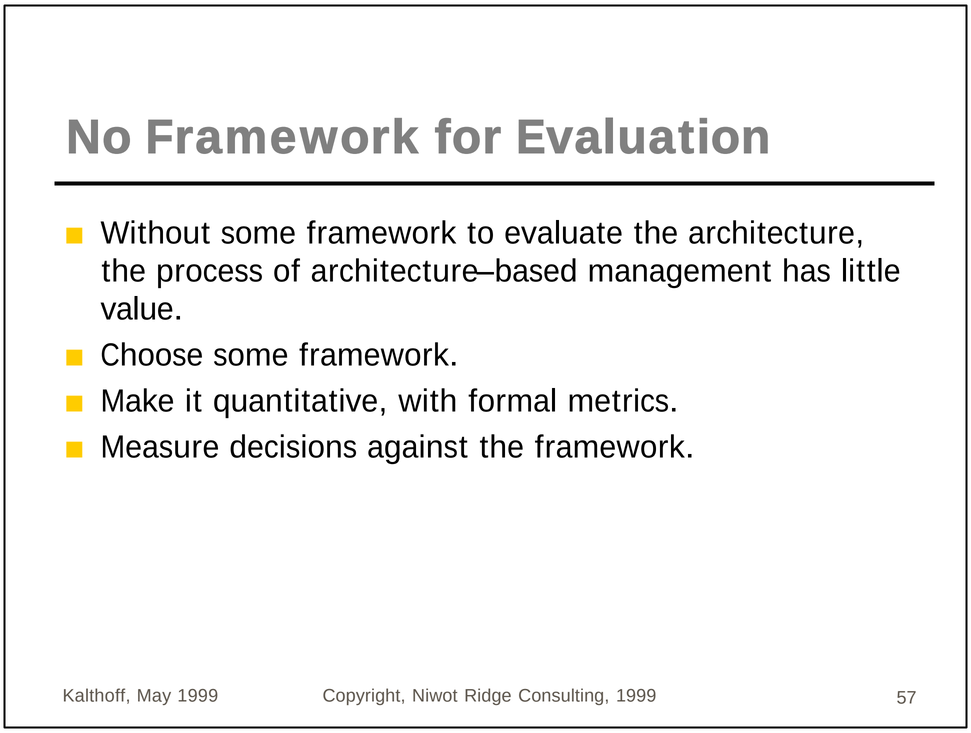


No Clear Description of the Functioning System

- The *Boxology* approach to system architecture
  - Ranges from a simple circle diagram of physical layers to complex boxes with lines connecting them
  - What are these lines and boxes?
    - Hardware and software components
    - Processes
    - Business activities
- System architecture notations are available, formalized and proven in the field – Use Them

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 56

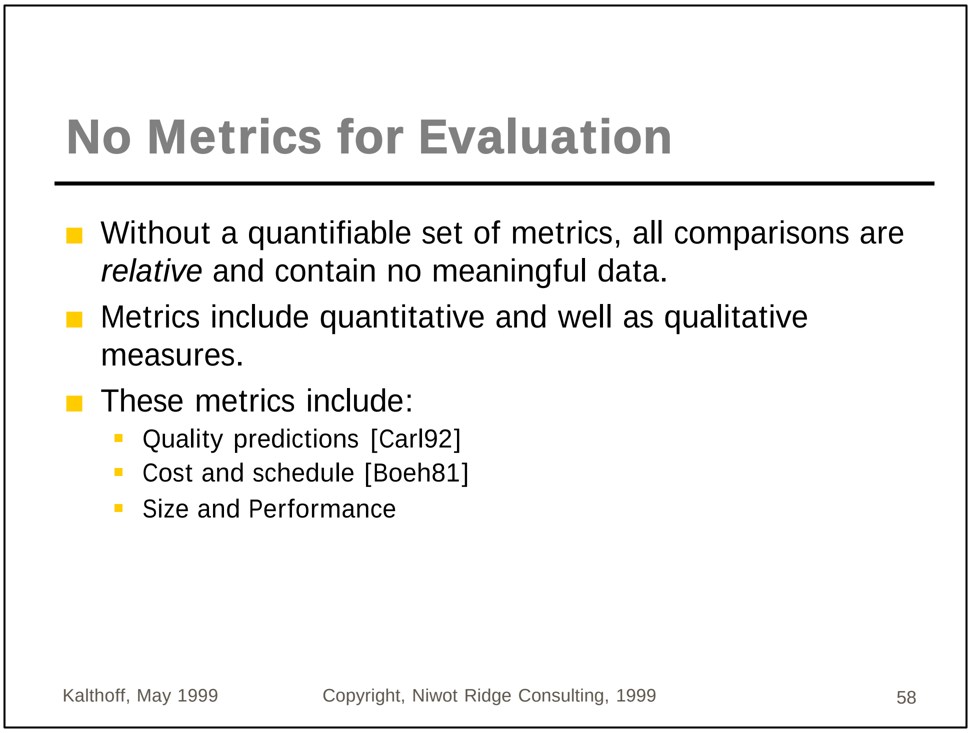
- Simply having boxes and lines does not provide a picture of the system. The lines and boxes are just ink on paper, they have not functionality, interoperability, or other behaviors.
- What needs to be developed is a picture of the system, possibly from multiple points of view, that can be used to *validate* the business case assumptions.
- This picture must be constructed using tools that allows analysis to be performed on the model.
  - That is, questions need to be asked and answered measured against various architectural tradeoffs.
  - The answers then need to be reviewed and consensus built to continue the validation process.



No Framework for Evaluation

- Without some framework to evaluate the architecture, the process of architecture–based management has little value.
- Choose some framework.
- Make it quantitative, with formal metrics.
- Measure decisions against the framework.

- Some domain must be provided by which the system architecture can be validated. Without this domain, the system is *theoretical* and of no practical use.

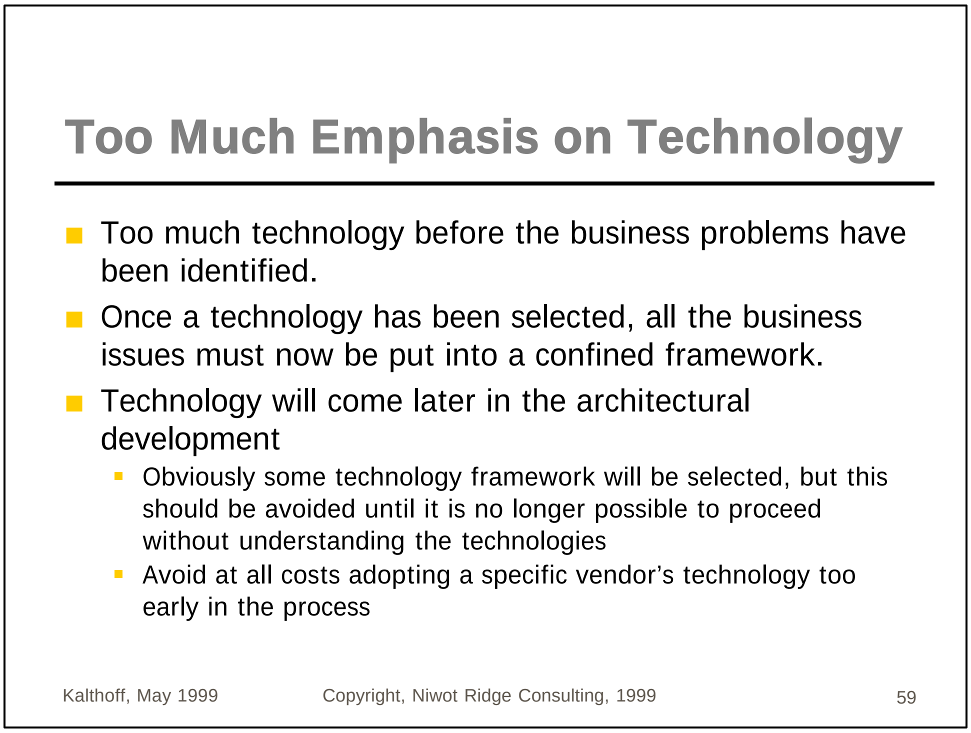


No Metrics for Evaluation

- Without a quantifiable set of metrics, all comparisons are *relative* and contain no meaningful data.
- Metrics include quantitative and well as qualitative measures.
- These metrics include:
  - Quality predictions [Carl92]
  - Cost and schedule [Boeh81]
  - Size and Performance

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 58

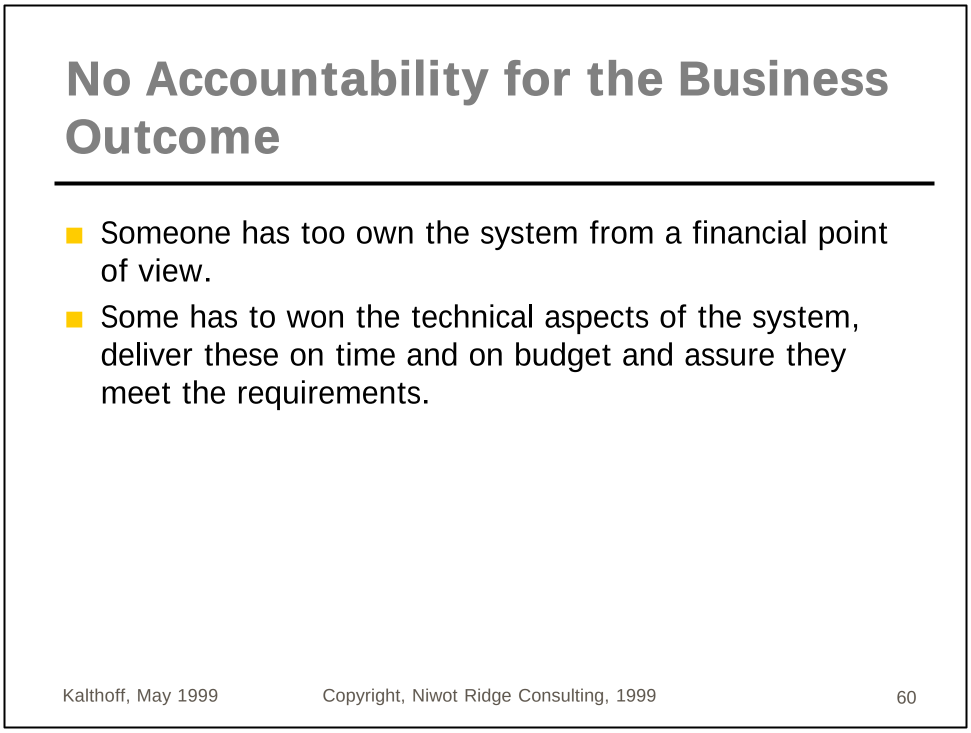
- When we're measuring something we need to know the units of measure, the coordinate systems and the scale of these coordinates systems.
- Without this information it's going to be difficult to determine any relative measure.
- How big? How much? How fast is fast? How fast is fast enough?
- And the dreaded words of the modern computer architecture *How do his scale*?


Too Much Emphasis on Technology

- Too much technology before the business problems have been identified.
- Once a technology has been selected, all the business issues must now be put into a confined framework.
- Technology will come later in the architectural development
  - Obviously some technology framework will be selected, but this should be avoided until it is no longer possible to proceed without understanding the technologies
  - Avoid at all costs adopting a specific vendor's technology too early in the process

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 59

- We're inundated with technology.
- From commercials for the latest smart car that practically drives itself, to software systems that somehow manage knowledge currently in the minds of the employee.
- The bottom line here is to put first things first - where is the business case for solving this problem.
- The question is not whether there is a problem, but is this problem worth solving?

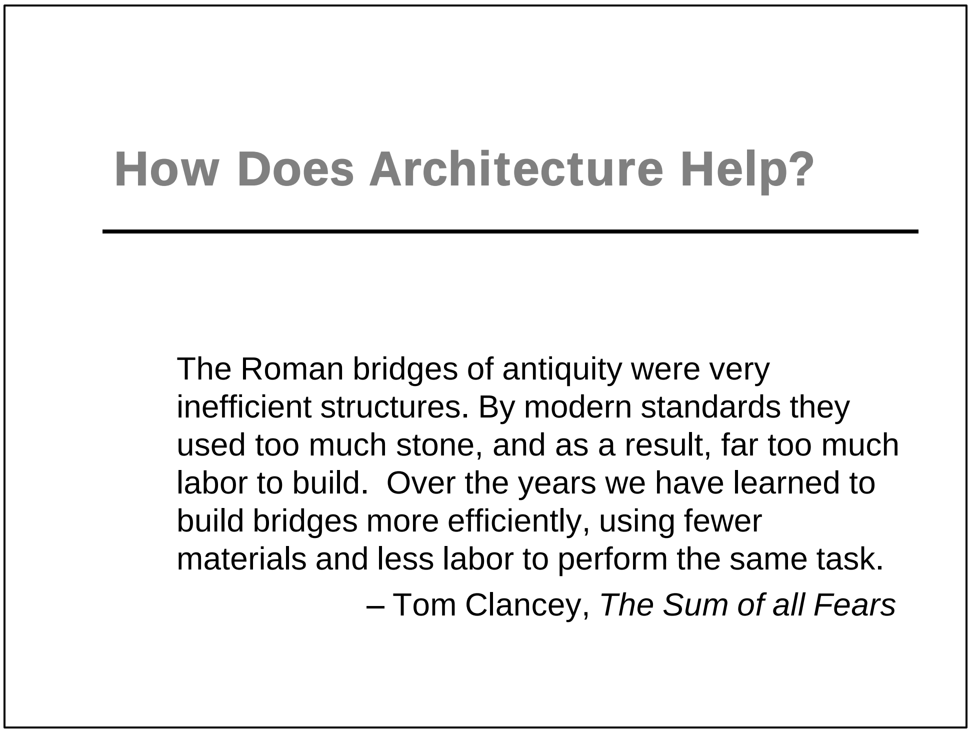


No Accountability for the Business Outcome

- Someone has too own the system from a financial point of view.
- Some has to won the technical aspects of the system, deliver these on time and on budget and assure they meet the requirements.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 60

- In the classic Brighman and Weston, Managerial Finance, the management role is to provide a single point of integrative responsibility.
- No management by committee, no management by consensus, no management by delegation - someone has to make the system work and the number come out right.

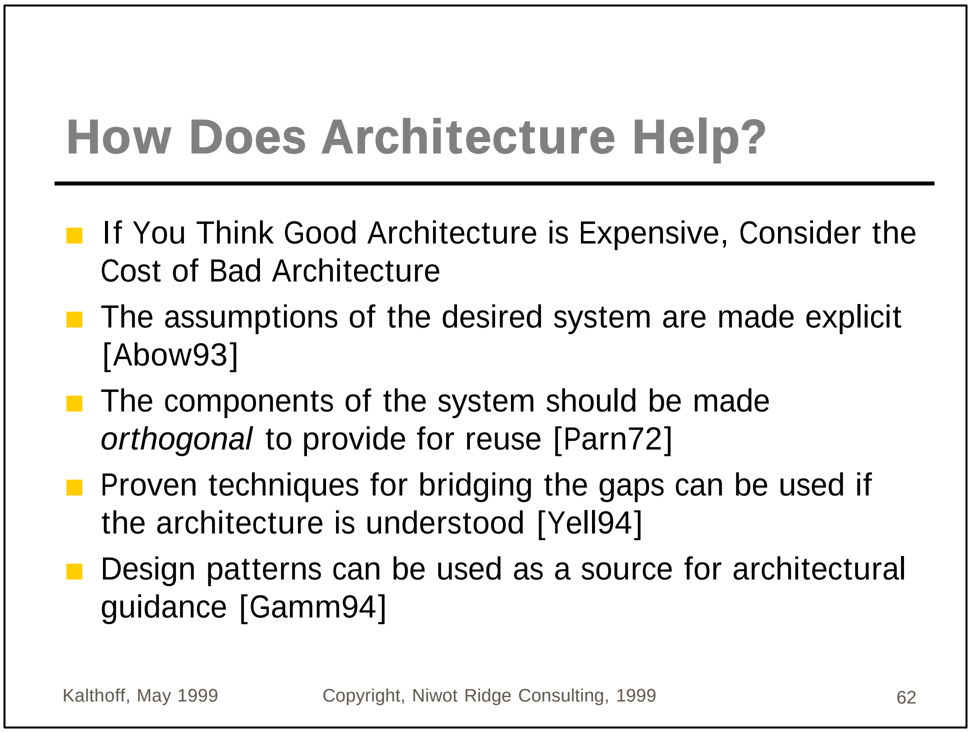


How Does Architecture Help?

The Roman bridges of antiquity were very inefficient structures. By modern standards they used too much stone, and as a result, far too much labor to build. Over the years we have learned to build bridges more efficiently, using fewer materials and less labor to perform the same task.

– Tom Clancey, *The Sum of all Fears*

- Without a mechanism for structuring the outcome, the probability of success is low - very low.

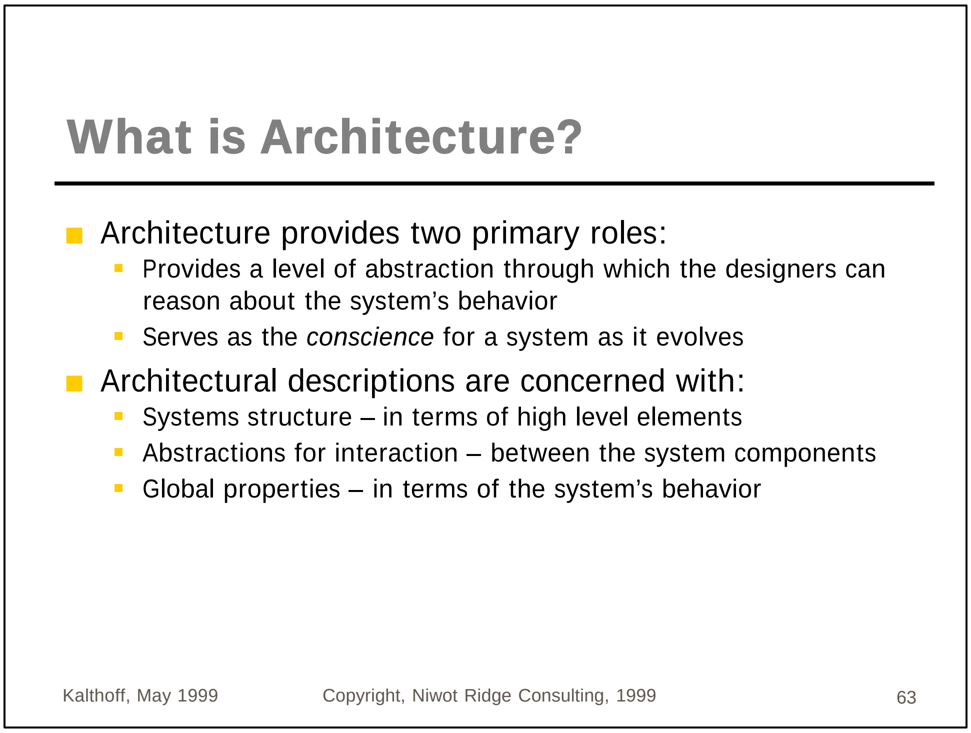


How Does Architecture Help?

- If You Think Good Architecture is Expensive, Consider the Cost of Bad Architecture
- The assumptions of the desired system are made explicit [Abow93]
- The components of the system should be made *orthogonal* to provide for reuse [Parn72]
- Proven techniques for bridging the gaps can be used if the architecture is understood [Yell94]
- Design patterns can be used as a source for architectural guidance [Gamm94]

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 62

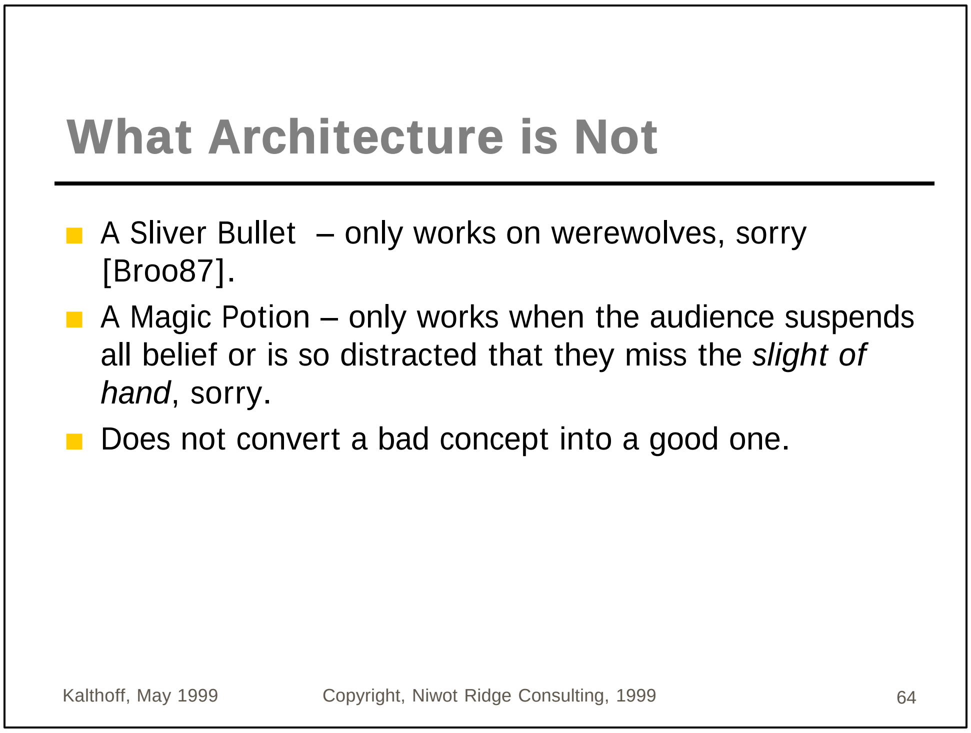
- Now that we come this far without asking any really hard questions, it's time for one.
- How does architecture help in any of these situations.
- The first answer may seem flippant, but it is one based on experience. In bad situations one approach to problem solving is to consider the alternatives and pick the one with the least consequences.


What is Architecture?

- Architecture provides two primary roles:
  - Provides a level of abstraction through which the designers can reason about the system's behavior
  - Serves as the *conscience* for a system as it evolves
- Architectural descriptions are concerned with:
  - Systems structure – in terms of high level elements
  - Abstractions for interaction – between the system components
  - Global properties – in terms of the system's behavior

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 63

- Thee are an unlimited number of definitions for architecture in the context of software systems, so I'll pick some of my favorites.
- One way to develop a definition is through an operational description -- what does architecture do?
- What is architecture concerned with?



What Architecture is Not

- A Sliver Bullet – only works on werewolves, sorry [Broo87].
- A Magic Potion – only works when the audience suspends all belief or is so distracted that they miss the *slight of hand*, sorry.
- Does not convert a bad concept into a good one.

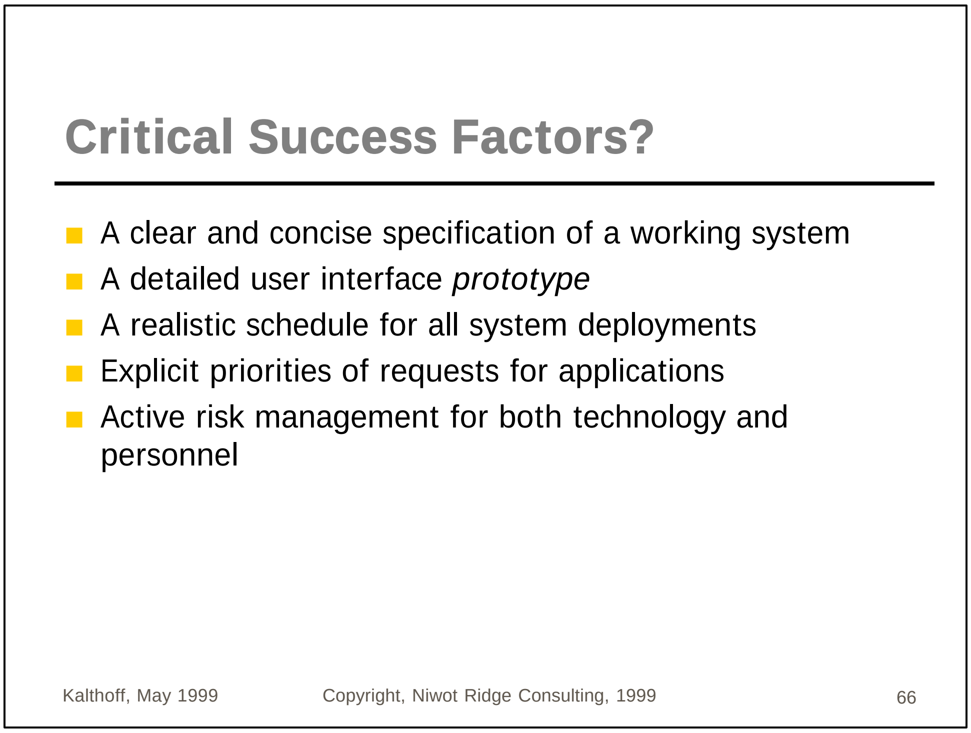
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 64

- Unfortunately for us there is no way out of the forest without blazing a trail.
- Architecture is one of the tools for moving projects forward
- it is one of those necessary but not sufficient activities
- Like creating fit, many thing have to come together
  - architecture
  - project management
  - business process engineering
  - financial management

How Does Architecture Make Money?

- Cost avoidance through standardization, business case deployment and strategic planning
- Enabling Information Technology deployment as a *strategic weapon*
- Resource utilization
- Agile *manufacturing* enabled through Information Technology

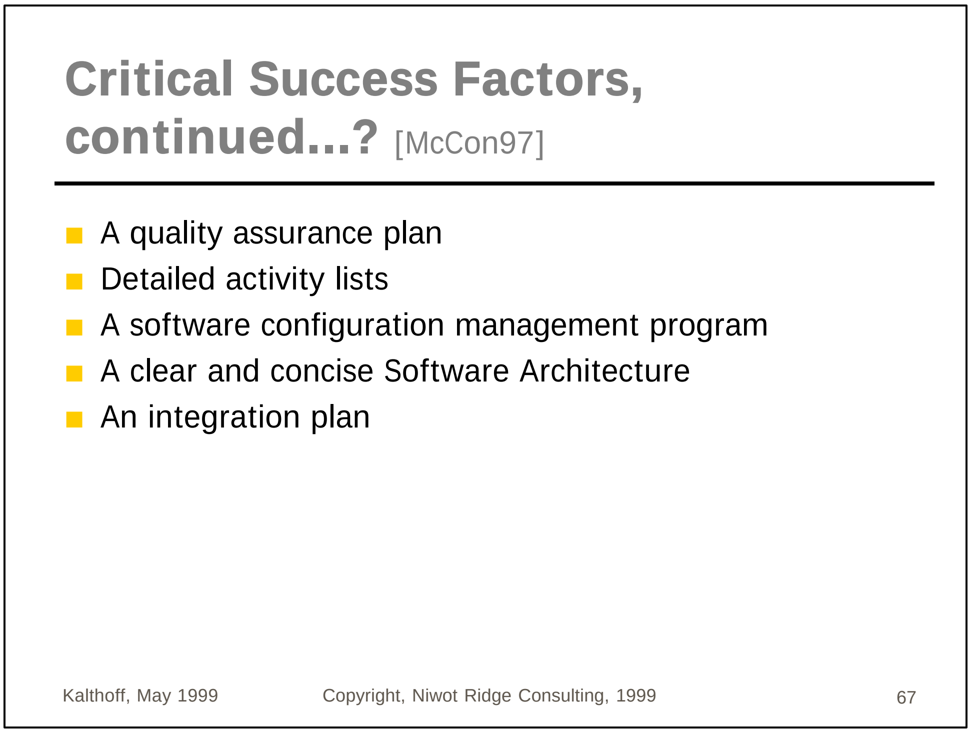
- So, how do we keep the finance folks off our back
- some tangible benefits to architectural efforts must be found
- The literature is good place to start as well as the archives of professional organizations.


Critical Success Factors?

- A clear and concise specification of a working system
- A detailed user interface *prototype*
- A realistic schedule for all system deployments
- Explicit priorities of requests for applications
- Active risk management for both technology and personnel

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 66

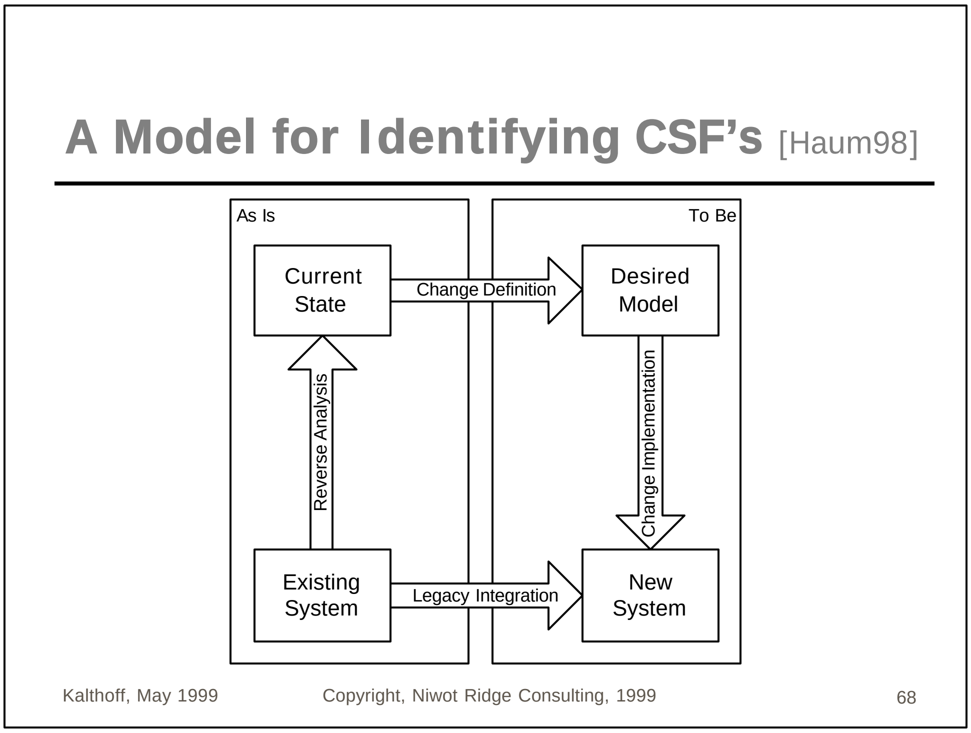
- Steve McConnel is a well know software development visionary. Steve writes a column in IEEE Software and several books on the subject of development
- His CSF's are a starting point for the Architecture Review Board
- The CSF concept was first presented by Rockhart. Everyone should find this paper and read it.



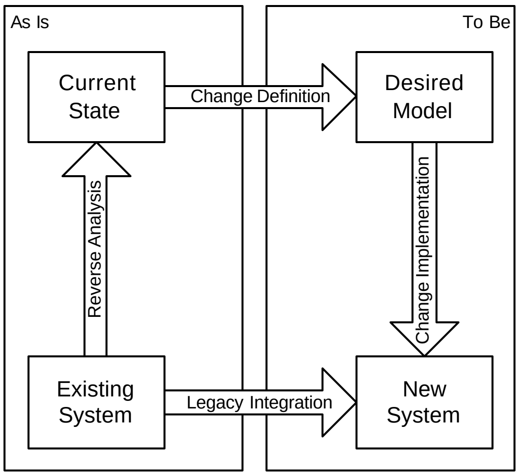
Critical Success Factors, continued...? [McCon97]

- A quality assurance plan
- Detailed activity lists
- A software configuration management program
- A clear and concise Software Architecture
- An integration plan

- Here are some more CSF's
- You should build your own CSF's, but read Rockhart's paper first to develop the background for writing these very important bullet lists.
- Just so we don't underestimate what's going on here, some of the CSF's are major development activities in themselves.

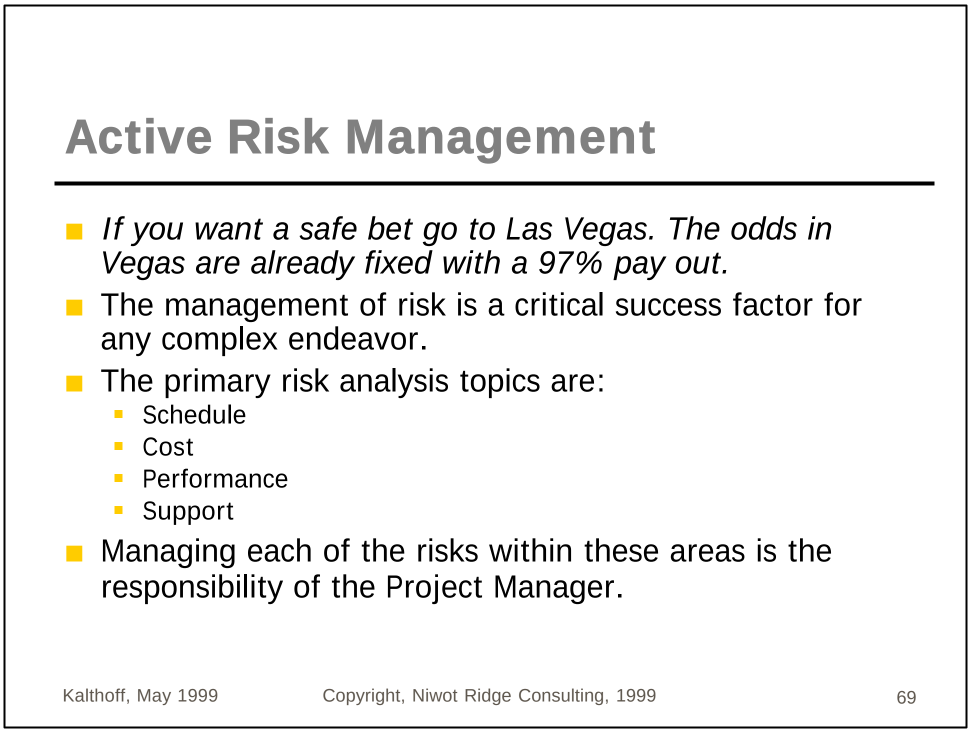


A Model for Identifying CSF's [Haum98]



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 68

- Here's one way to structure the CSF's around a project.
- Each of the CSF's can be measured against four states of the system:
  - Reverse analysis - defining the current state model. There is usually not one.
  - Change definition - defining the new model
  - Change implementation - designing, implementing, testing and installing the new system.
  - Legacy integration - considering the existing context when performing all the above activities.

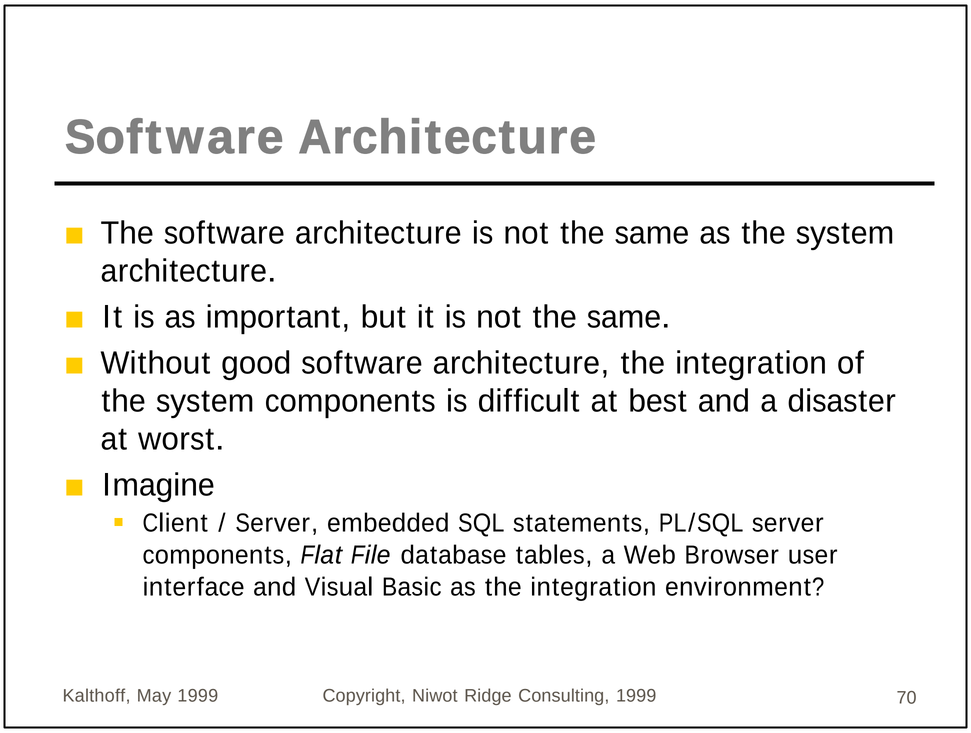


Active Risk Management

- *If you want a safe bet go to Las Vegas. The odds in Vegas are already fixed with a 97% pay out.*
- The management of risk is a critical success factor for any complex endeavor.
- The primary risk analysis topics are:
  - Schedule
  - Cost
  - Performance
  - Support
- Managing each of the risks within these areas is the responsibility of the Project Manager.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 69

- Many of the CSF's are obvious or can be looked up in books.
- The Risk Management components are not so obvious.

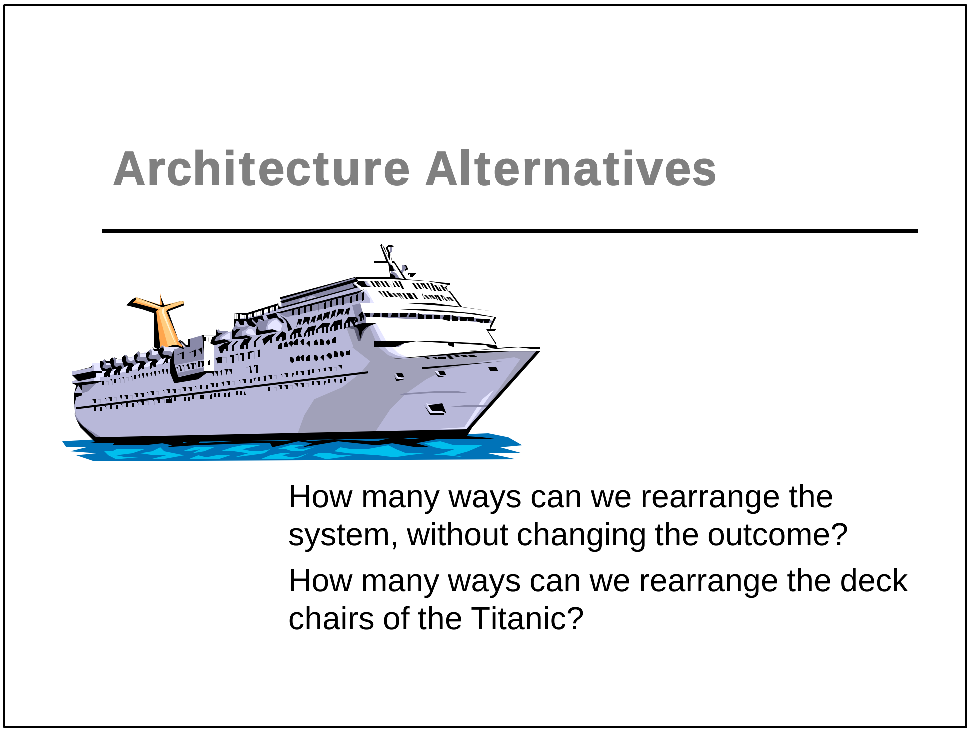


Software Architecture

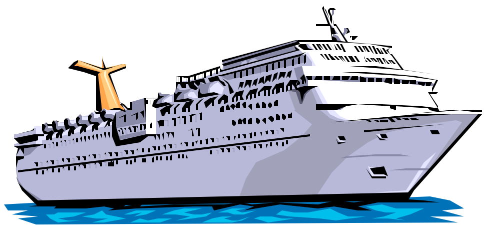
- The software architecture is not the same as the system architecture.
- It is as important, but it is not the same.
- Without good software architecture, the integration of the system components is difficult at best and a disaster at worst.
- Imagine
  - Client / Server, embedded SQL statements, PL/SQL server components, *Flat File* database tables, a Web Browser user interface and Visual Basic as the integration environment?

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 70

- The difference between software architecture and system architecture is one of domain.
- The software architecture describes how the internal components of the system behave.
- The system architecture describes how the components interact.
- The description in the last bullet is not too far from the truth.



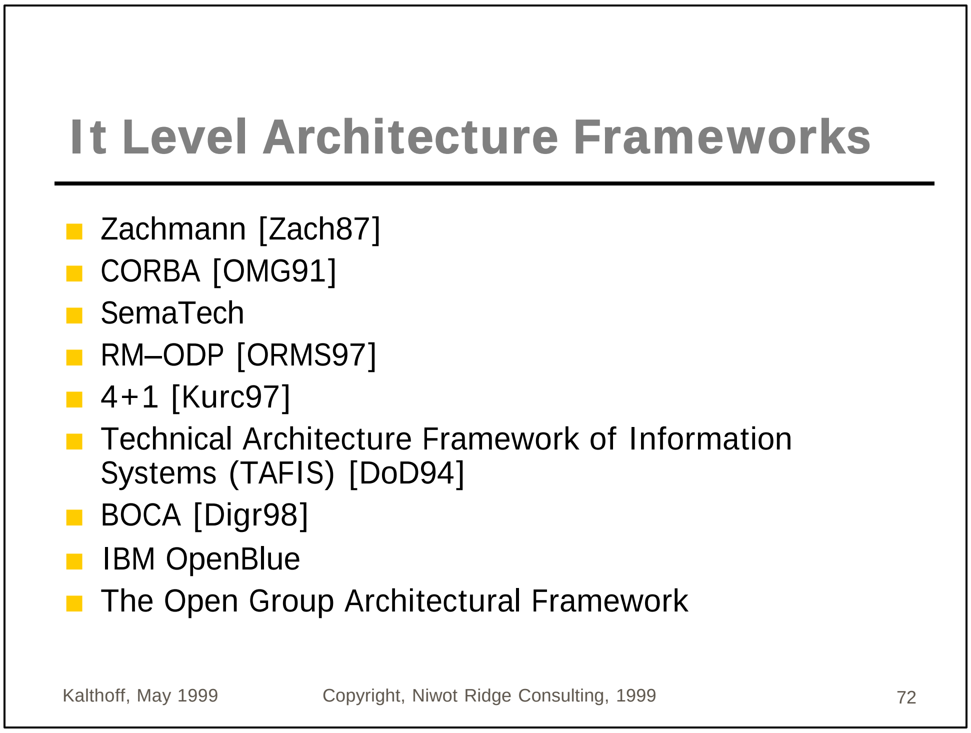
Architecture Alternatives



How many ways can we rearrange the system, without changing the outcome?

How many ways can we rearrange the deck chairs of the Titanic?

- One good place to find good architecture is to look at published architectures.

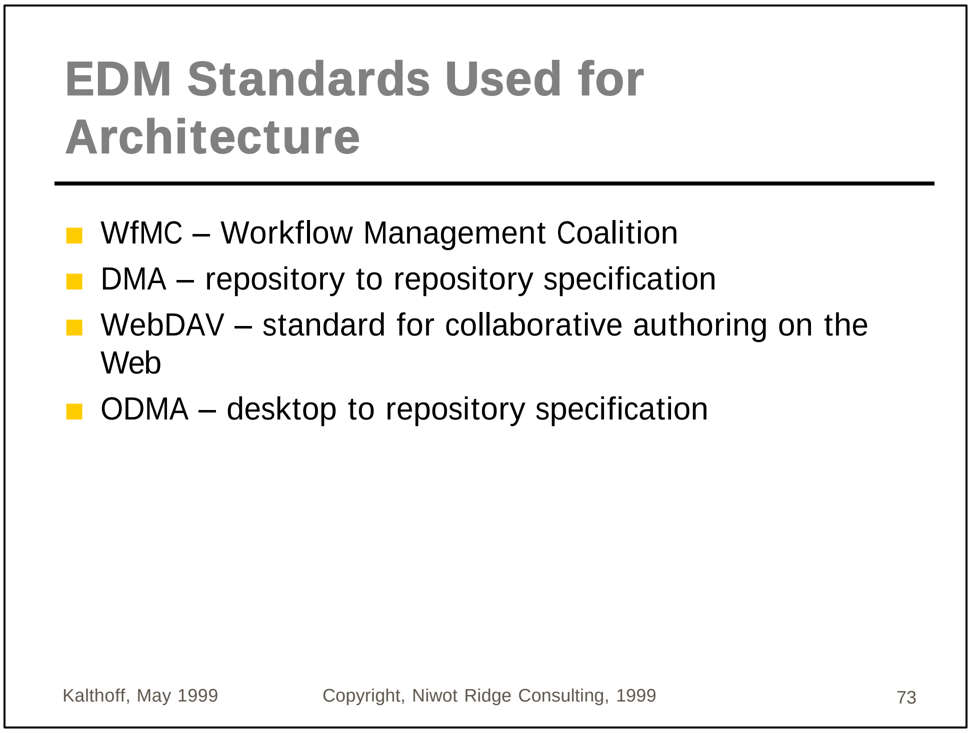


It Level Architecture Frameworks

- Zachmann [Zach87]
- CORBA [OMG91]
- SemaTech
- RM–ODP [ORMS97]
- 4+1 [Kurc97]
- Technical Architecture Framework of Information Systems (TAFIS) [DoD94]
- BOCA [Digr98]
- IBM OpenBlue
- The Open Group Architectural Framework

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 72

- Here is a short list of architectures and the vendors or agencies that maintain them.

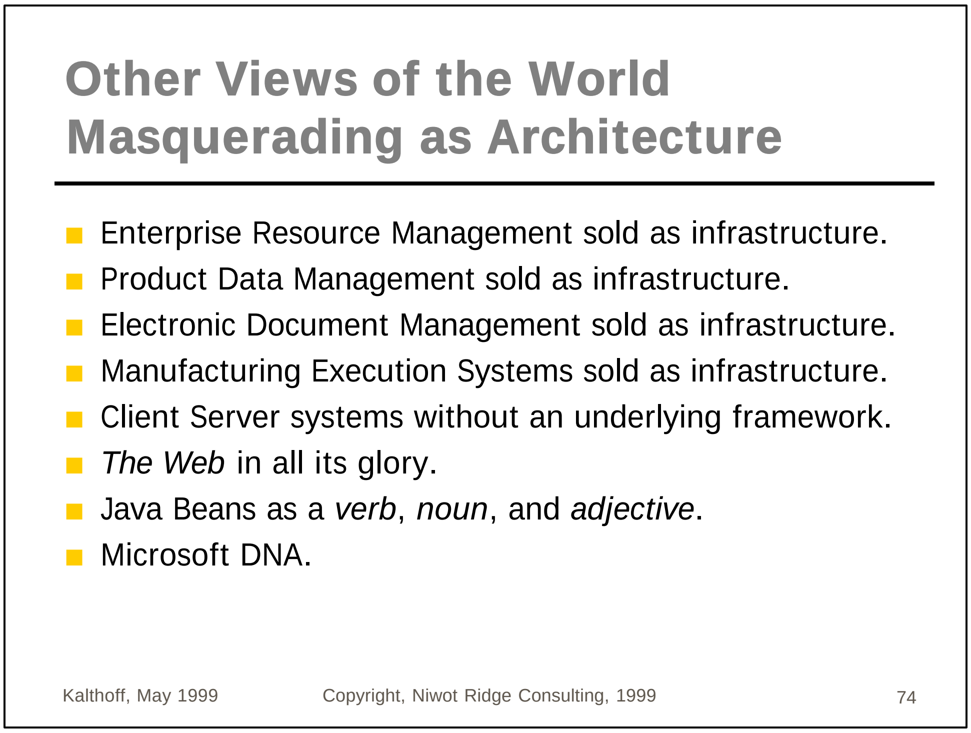


EDM Standards Used for Architecture

- WfMC – Workflow Management Coalition
- DMA – repository to repository specification
- WebDAV – standard for collaborative authoring on the Web
- ODMA – desktop to repository specification

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 73

- In addition to the layered architectures in the previous slide, the EDM world has some specific standards that provide physical architecture.


Other Views of the World Masquerading as Architecture

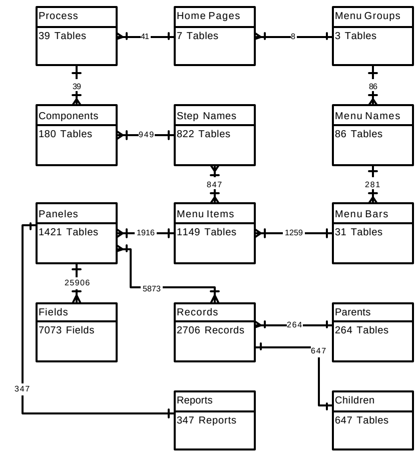
- Enterprise Resource Management sold as infrastructure.
- Product Data Management sold as infrastructure.
- Electronic Document Management sold as infrastructure.
- Manufacturing Execution Systems sold as infrastructure.
- Client Server systems without an underlying framework.
- *The Web* in all its glory.
- Java Beans as a *verb*, *noun*, and *adjective*.
- Microsoft DNA.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 74

- Since the definition of architecture and the practice of architecture are poorly defined, there is an open business opportunity for vendors to make their own definitions.
- I've been in meetings where the architecture of a product is being presented and the picture on the screen was a wire connected to boxes.
- I've been in other meetings where the architecture was described in terms of the number of applications programming interfaces available.
- Architecture is none of these.
- The architecture we're interested in is based on the enablement of the business process.

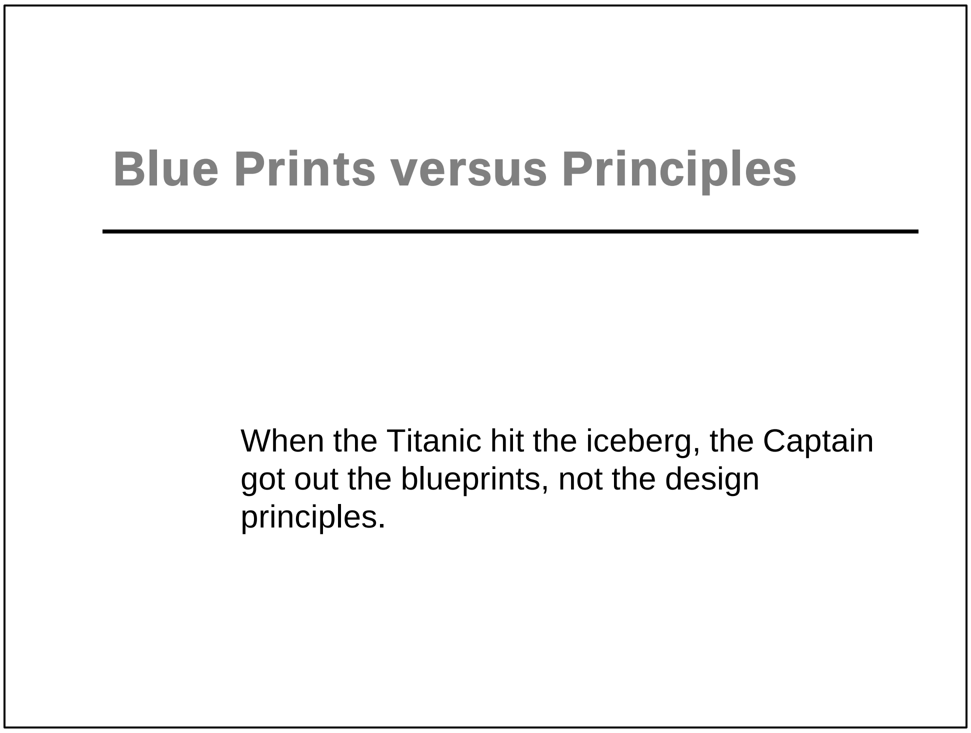
A *Simple* View of an ERP System

[Aike99]



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 75

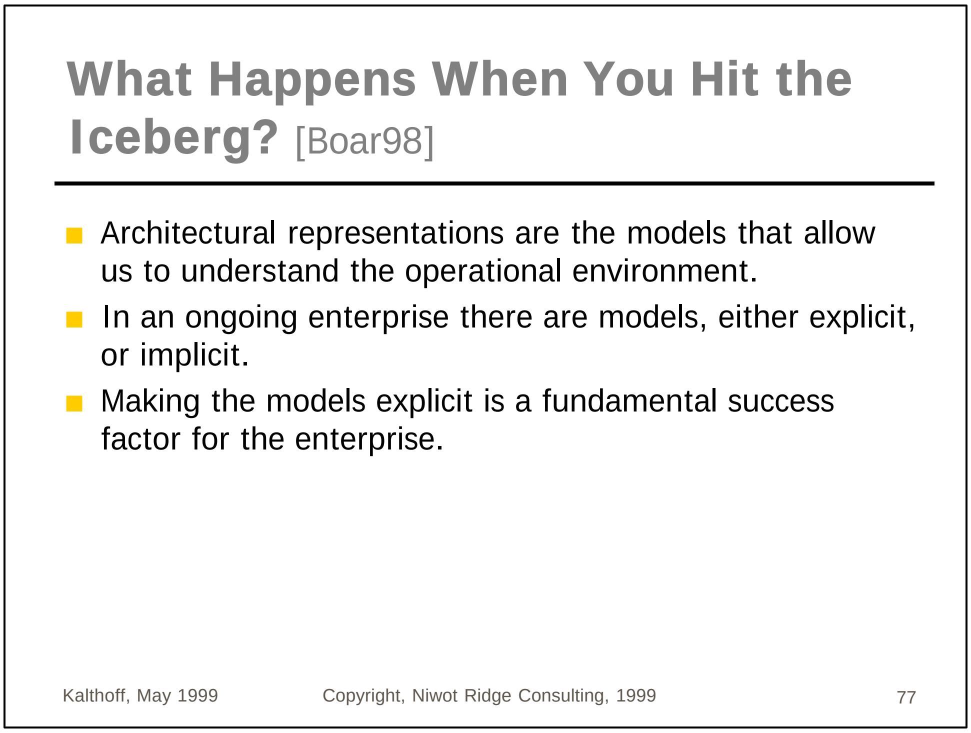
- Here's a simple example of why integrating the ERP paradigm with other systems, runs into trouble at the first turn.
- This the the metadata model of a popular ERP system.



Blue Prints versus Principles

When the Titanic hit the iceberg, the Captain got out the blueprints, not the design principles.

- Now let’s separate the concept of plans from principles.

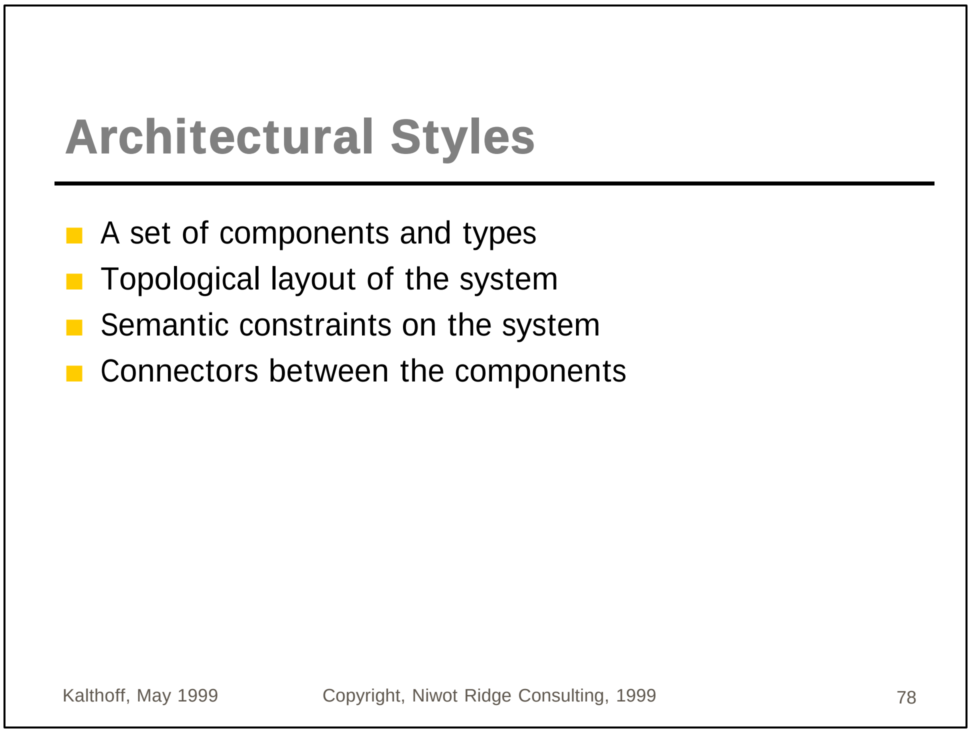


What Happens When You Hit the Iceberg? [Boar98]

- Architectural representations are the models that allow us to understand the operational environment.
- In an ongoing enterprise there are models, either explicit, or implicit.
- Making the models explicit is a fundamental success factor for the enterprise.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 77

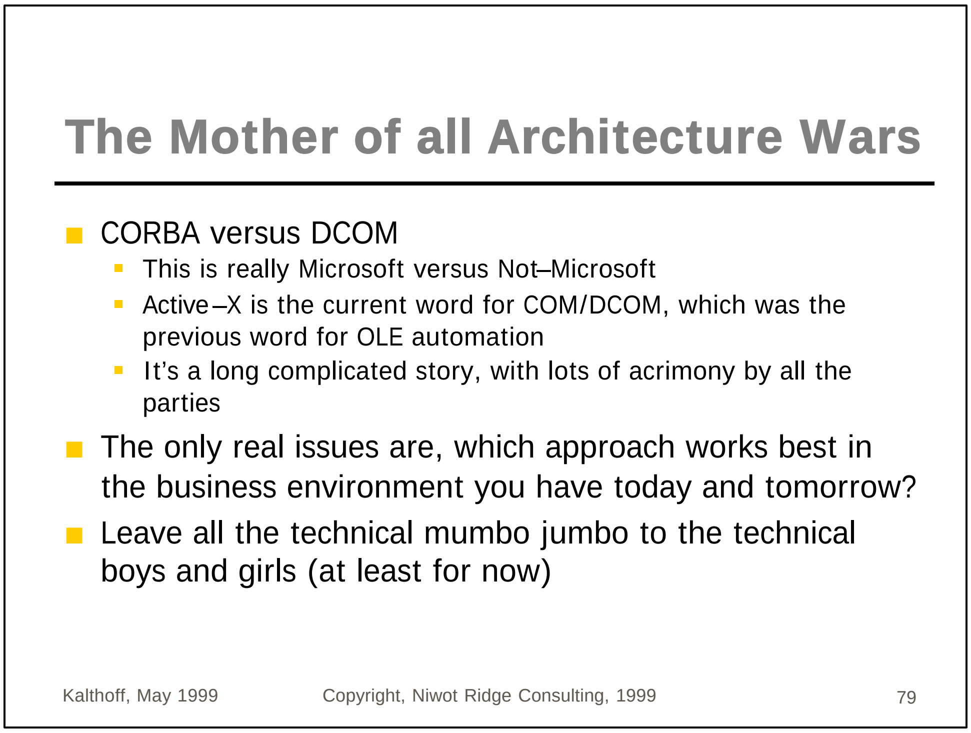
- This concept of using blueprints instead of principles is important to understanding the role of architecture.
- The results of architecture is the plans for the system.



Architectural Styles

- A set of components and types
- Topological layout of the system
- Semantic constraints on the system
- Connectors between the components

- The style of a system is defined by these attributes.
- This is an area of research that has made good progress in recent years.
- This may seem very esoteric, but if we are going to protect ourselves from the marketing guys, then some form of taxonomy is needed to sort out the marketecture from the architecture.

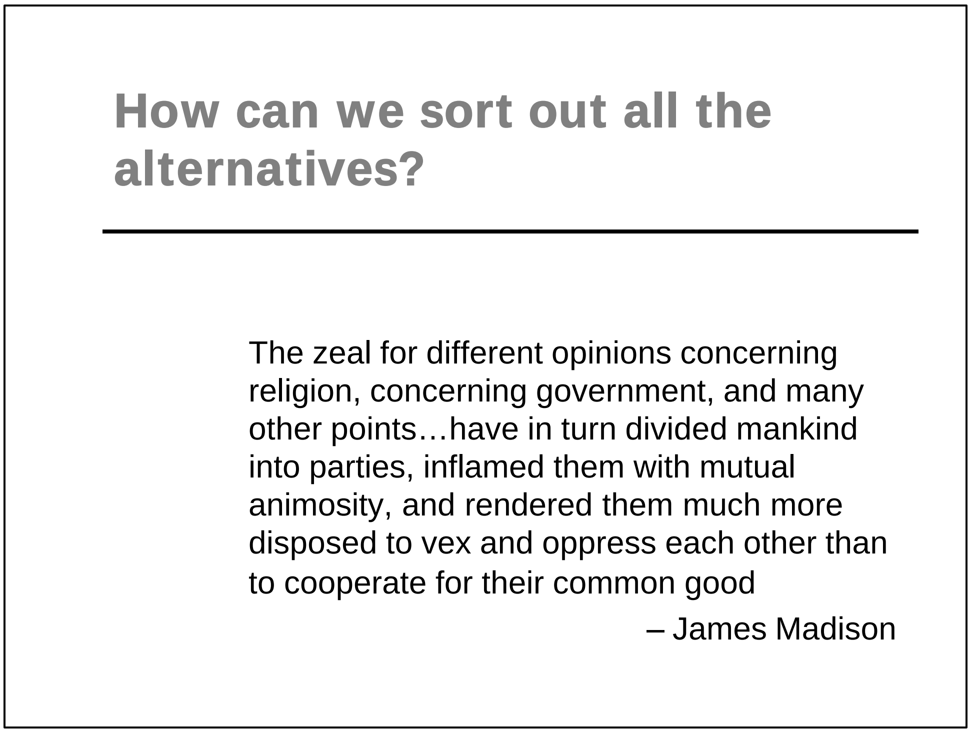


The Mother of all Architecture Wars

- CORBA versus DCOM
  - This is really Microsoft versus Not–Microsoft
  - Active–X is the current word for COM/DCOM, which was the previous word for OLE automation
  - It's a long complicated story, with lots of acrimony by all the parties
- The only real issues are, which approach works best in the business environment you have today and tomorrow?
- Leave all the technical mumbo jumbo to the technical boys and girls (at least for now)

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 79

- I really want to avoid the theological issues here.
- The market place will sort this out over time.
- In all likelihood there will be two standards, the Microsoft way of doing things and the Not–Microsoft way of doing things.

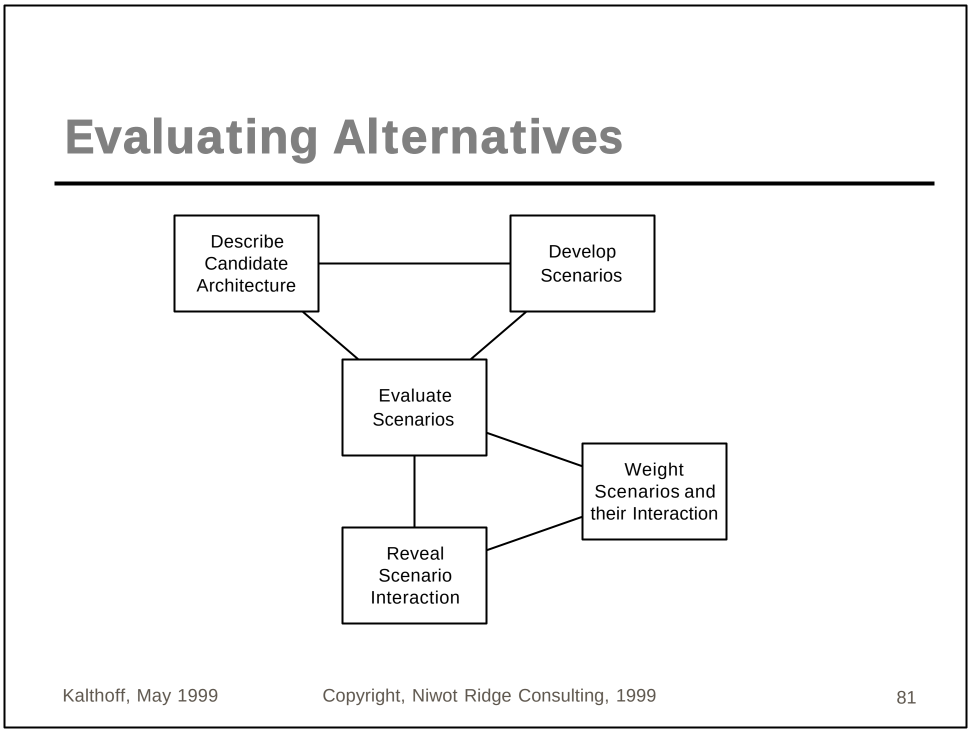


How can we sort out all the alternatives?

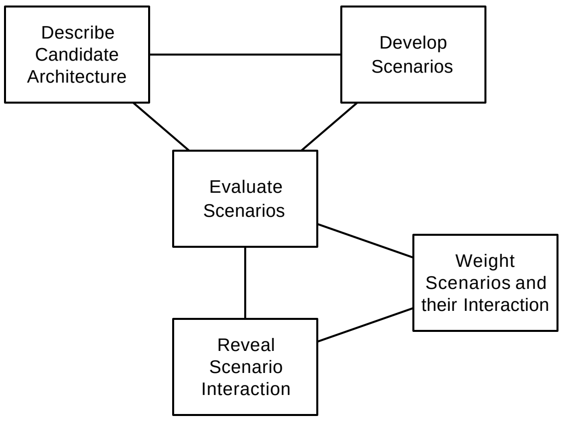
The zeal for different opinions concerning religion, concerning government, and many other points…have in turn divided mankind into parties, inflamed them with mutual animosity, and rendered them much more disposed to vex and oppress each other than to cooperate for their common good

– James Madison

- Madison’s words are as meaningful today as they were in the Federalist Paper Number 10, presented November 23, 1787.
- We must allow ourselves to step back and examine the various options, without introducing the current marketecture into the project.

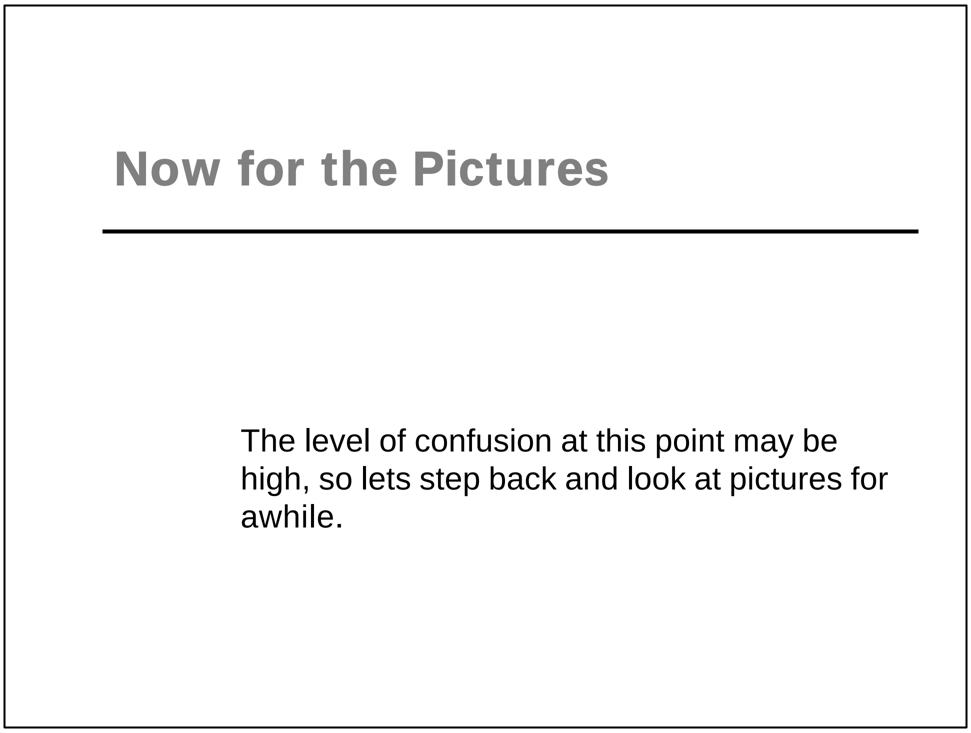


Evaluating Alternatives


Describe Candidate Architecture
Develop Scenarios
Evaluate Scenarios
Weight Scenarios and their Interaction
Reveal Scenario Interaction

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 81

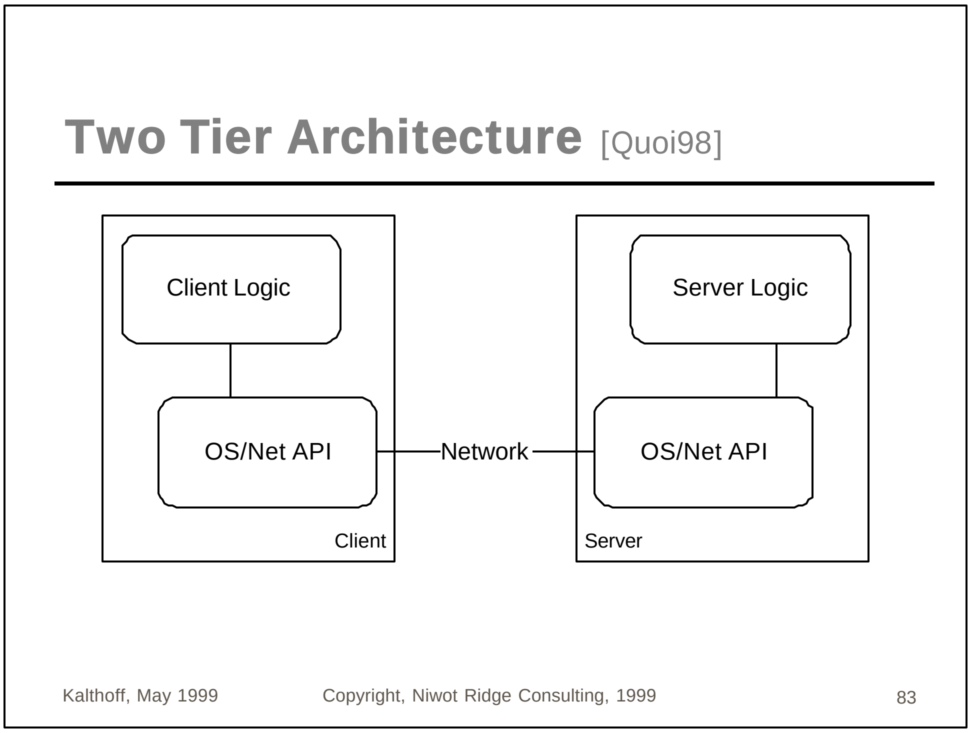
- One approach to sorting out all the alternatives is to have an iterative process shown here.
- This is pretty simple and represents a very high level view of the evaluation process.
- An actual process should have:
  - Evaluation metrics.
  - Evaluation guidelines for each component being compared.



Now for the Pictures

The level of confusion at this point may be high, so lets step back and look at pictures for awhile.

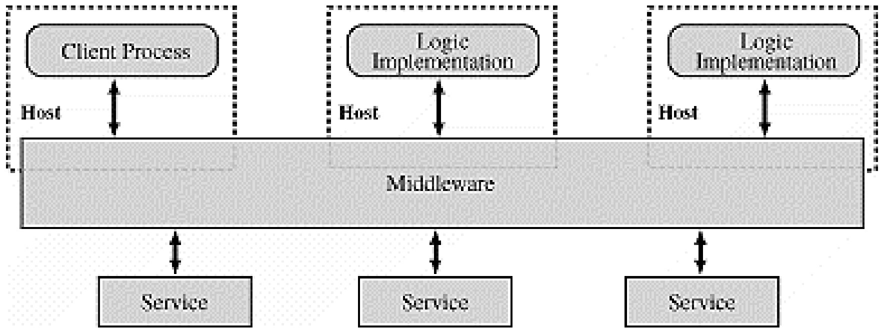
- There always have to be pictures in the presentation, but this subject does not lend itself well to pictures. So remember this when you fill out the session evaluation...there were pictures in the presentation.



Two Tier Architecture [Quoi98]

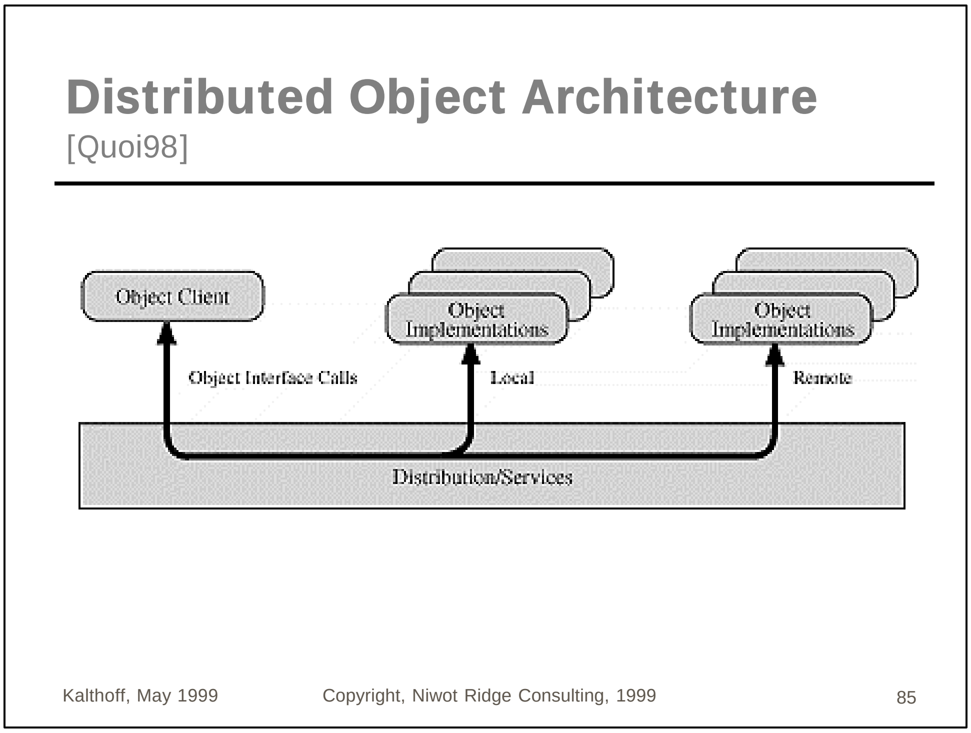
- The two tier architecture is based on a remote procedure call mechanism that is hosted across the network.
- Some form of API is provided between the client and the server. This can vary between vendors, but the most popular today is based on TCP/IP packets.
- But having TCP/IP is only the beginning. This is usually called the wire protocol.
- In the next slide we’ll see more layers and more confusion.

Three Tier Architecture [Quoi98]



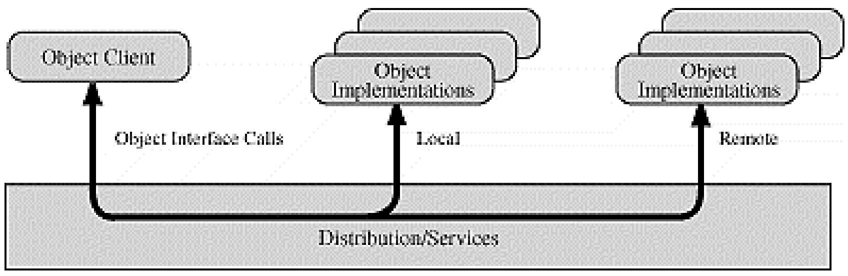
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 84

- In the Three Tier architecture a client interacts with a service through a *Middleware* application.
- The Three Tier nomenclature has been evolved to *Middleware* by the Object Management Group.
- This new layer allows clients to interact with a generic abstraction of a server rather than with a specific host and/or process.
- These abstractions are provided through Application Programming Interfaces without knowledge of the location or implementation of the external facility.



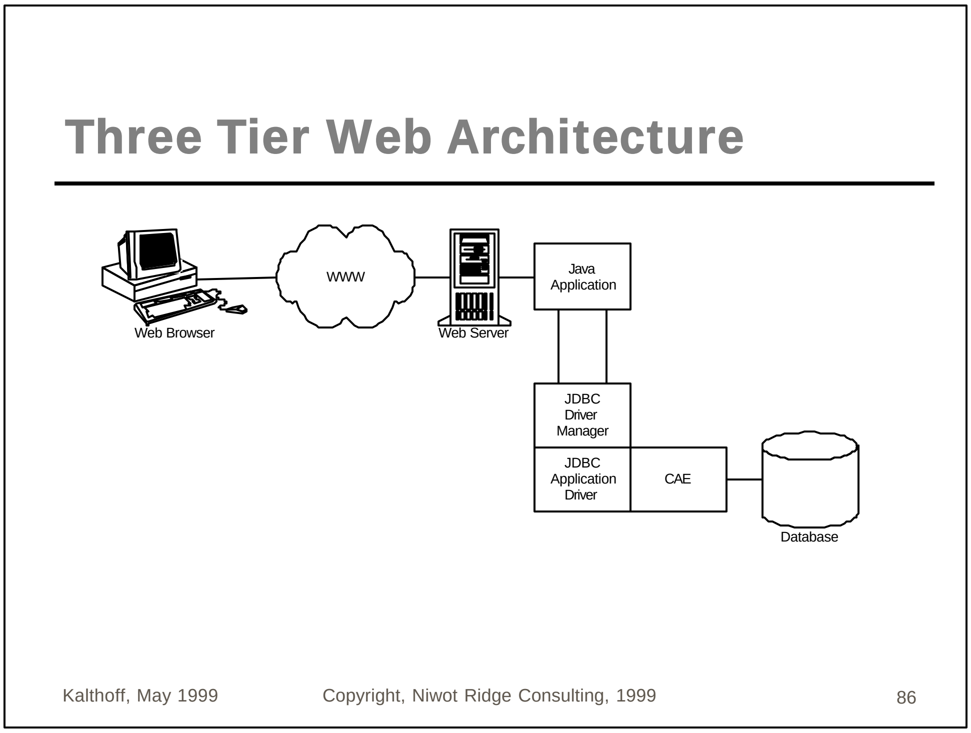
Distributed Object Architecture

[Quoi98]

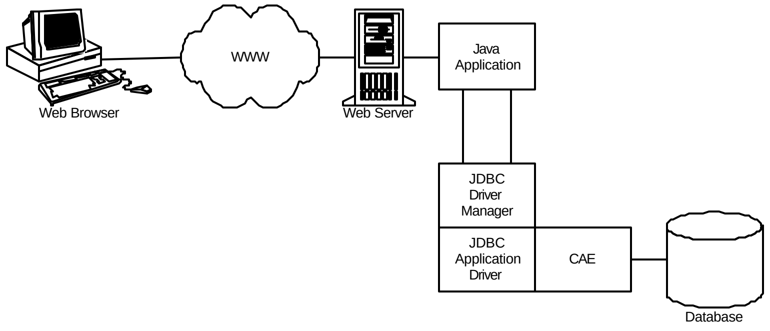


Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 85

- The distributed object world revolves around CORBA and DCOM. In both cases objects communicate with each other through some form of distribution service.
- An interface for the distribution service is made available to all the participants.
- Once a connection is established (not a connection in terms of a phone line, but a service connection) the client can make requests of the service.



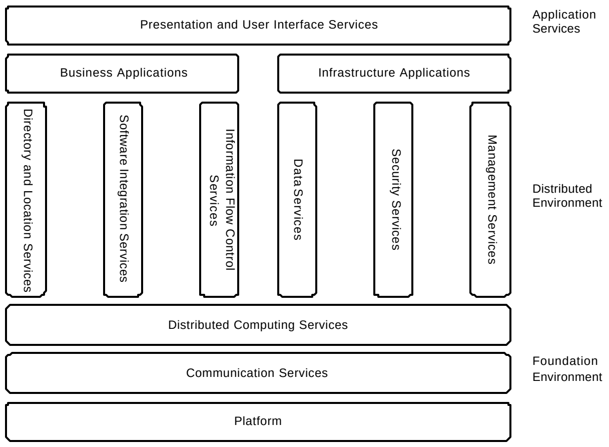
Three Tier Web Architecture



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 86

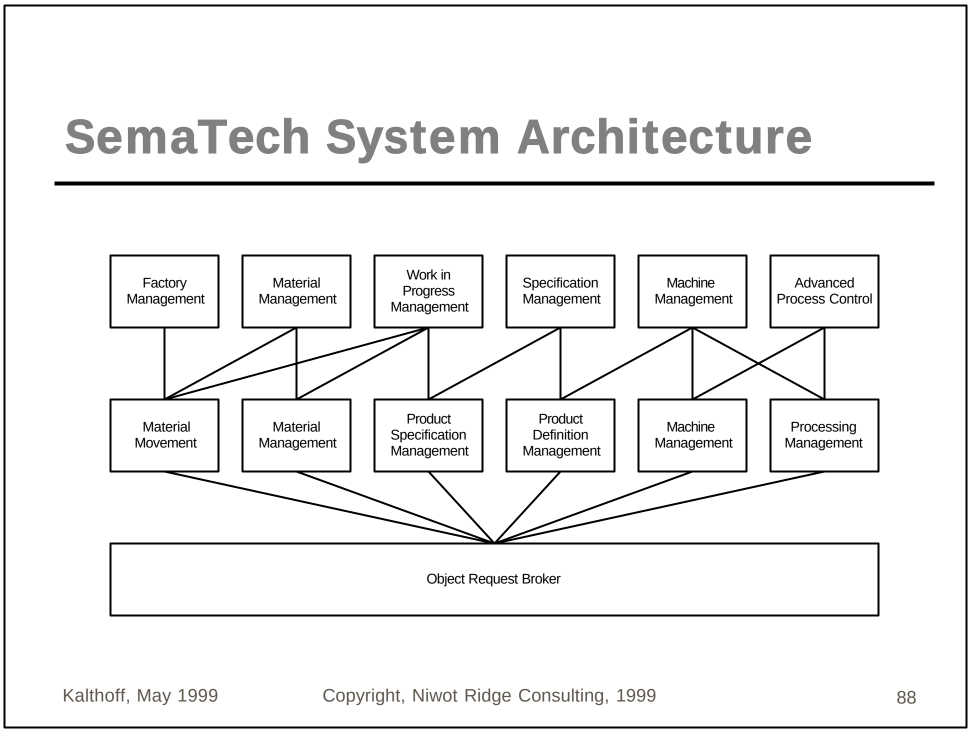
- In the web world the concept of three tier is nearly the same, only the components have different names.
- I'll take the Java application point of view. And this is a simplified view.
- The client (tier 1) runs a standard web browser.
- The middle tier (tier 2) runs Java applications.
- And the third tier is the database engine.
- This very simple picture is always more complex in practice. With Java Beans, Application and Server side adapters for the various services, database transaction processors and the like.

The Open Group Architectural Framework [Toga99]

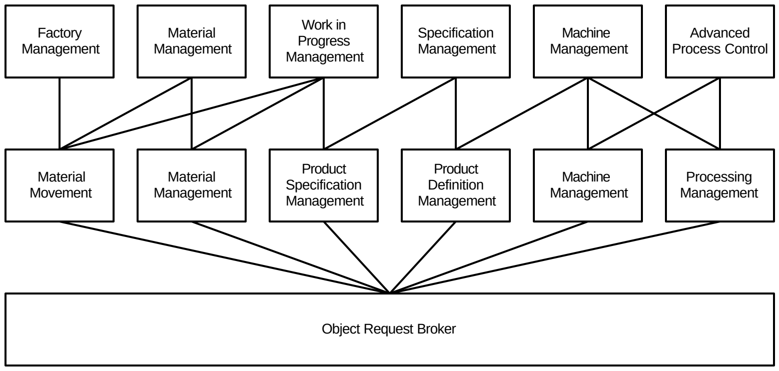

Presentation and User Interface Services
Application Services
Business Applications
Infrastructure Applications
Directory and Location Services
Software Integration Services
Information Flow Control Services
Data Services
Security Services
Management Services
Distributed Environment
Distributed Computing Services
Communication Services
Foundation Environment
Platform

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 87

- Another type of architecture is the original Open System Foundation OSF/1 structure.
- This is the architecture of large client server applications in the Sun Solaris world (primarily).



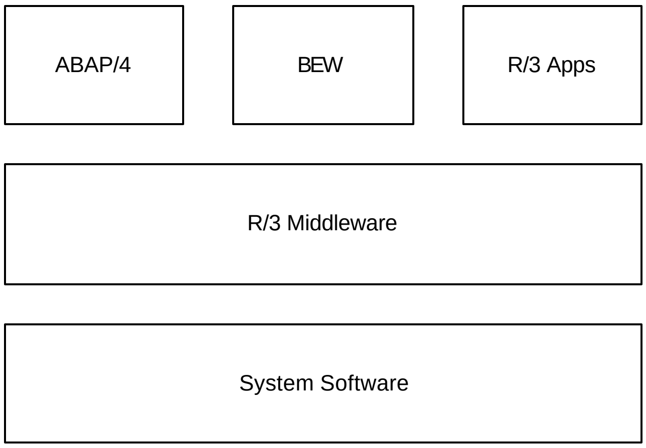
SemaTech System Architecture



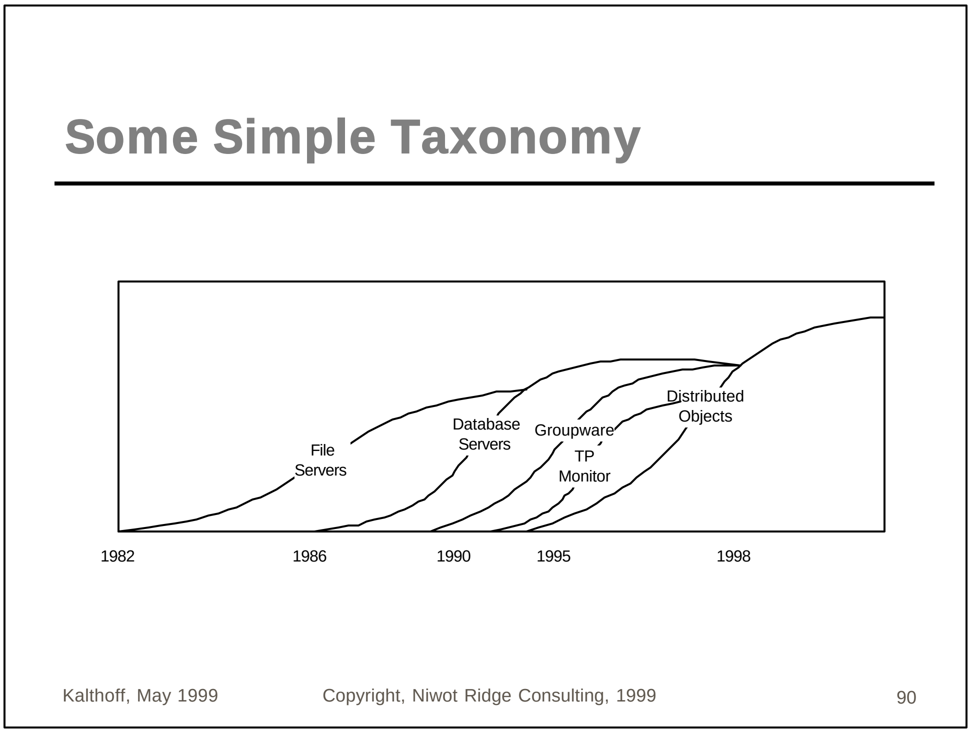
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 88

- The SemaTech architecture makes use of Corba processes to integrate the various business processes.

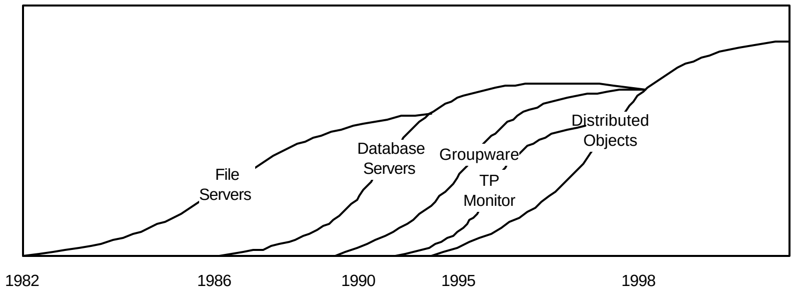
SAP R/3 High Level [Busk96]



- In the SAP architecture a straight forward three tier client server model is used.

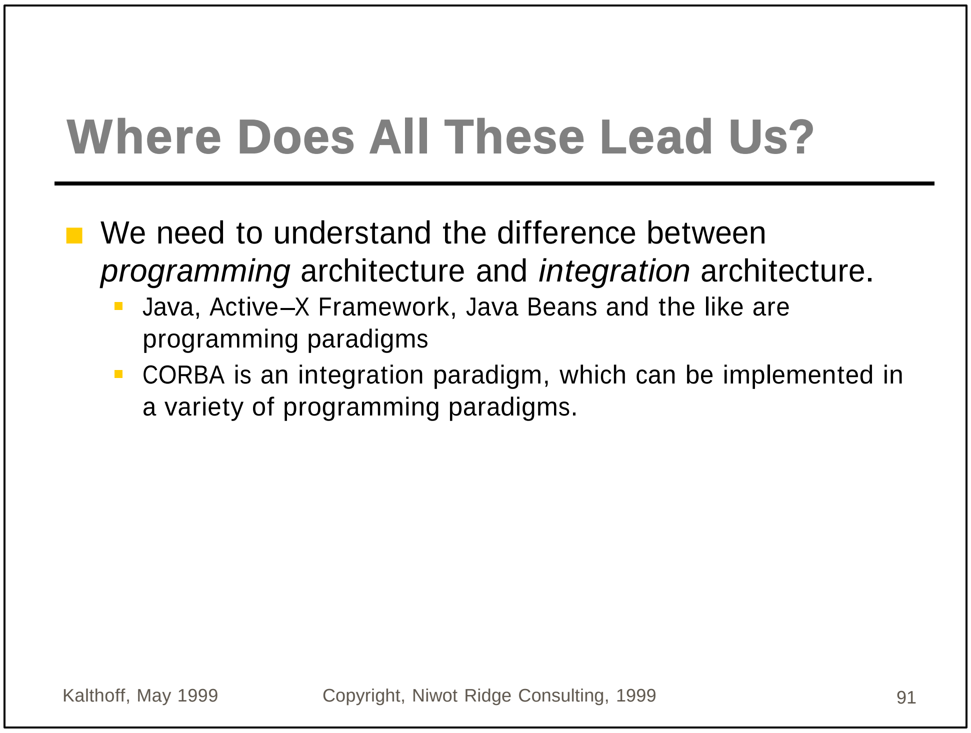


Some Simple Taxonomy



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 90

- Understanding how we came to where we are is a nice diversion after all this technical hand waving.
- Here's the standard mountain wave graph for the distributed object world. The web is not in here because it has actually overtaken most of these waves and will continue to do so in the application environments that support user interfaces.

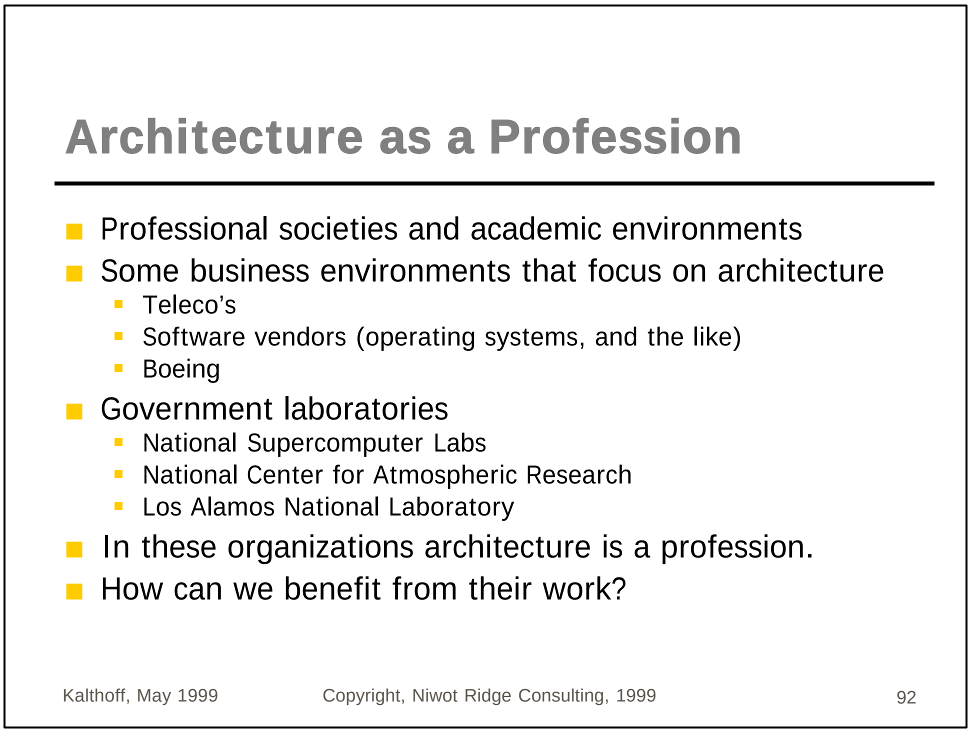


Where Does All These Lead Us?

- We need to understand the difference between *programming* architecture and *integration* architecture.
  - Java, Active–X Framework, Java Beans and the like are programming paradigms
  - CORBA is an integration paradigm, which can be implemented in a variety of programming paradigms.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 91

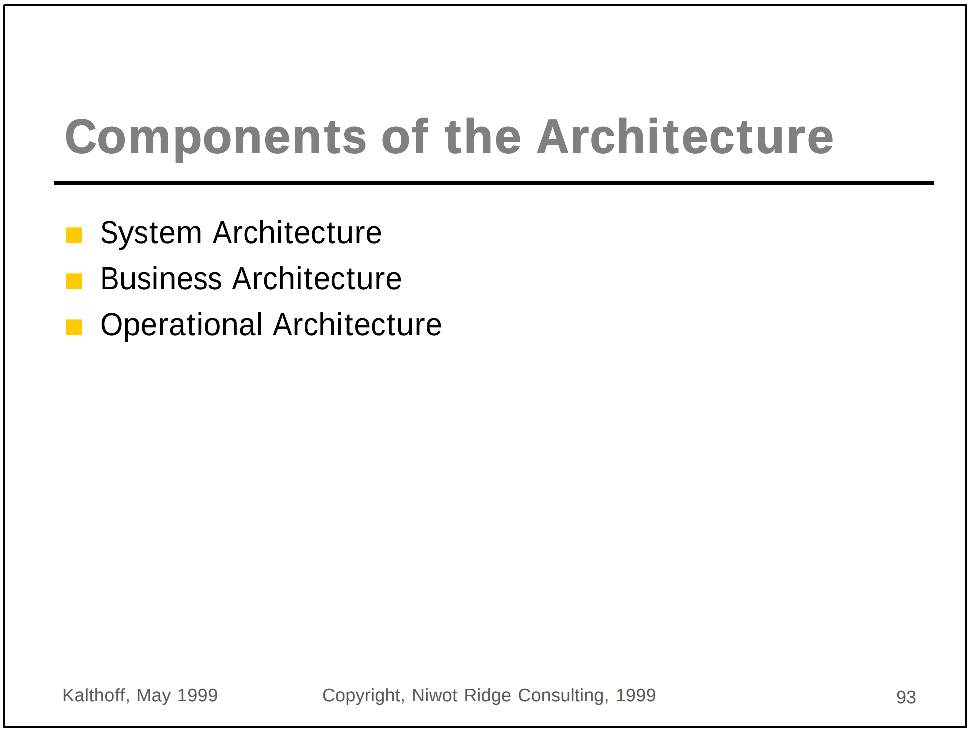
- All of these pictures share one common flaw. They are descriptions of how a specific software vendor, Sun, Microsoft, IBM, etc. see the system architecture.
- In reality they are architectures for programming the system components. They speak nothing about the business process architecture and how it is represented in the system.



Architecture as a Profession

- Professional societies and academic environments
- Some business environments that focus on architecture
  - Teleco's
  - Software vendors (operating systems, and the like)
  - Boeing
- Government laboratories
  - National Supercomputer Labs
  - National Center for Atmospheric Research
  - Los Alamos National Laboratory
- In these organizations architecture is a profession.
- How can we benefit from their work?

- Since architecture is one of the buzz words like, Java, the Web, object oriented, many vendors, marketing departments and heaven forbid consultants have cooped the word for their own purposes.
- Architecture and marketecture have become interchangeable.
- Lets' not get confused though by the marketecture folks. Architecture is a profession practiced in many areas of the computer business.

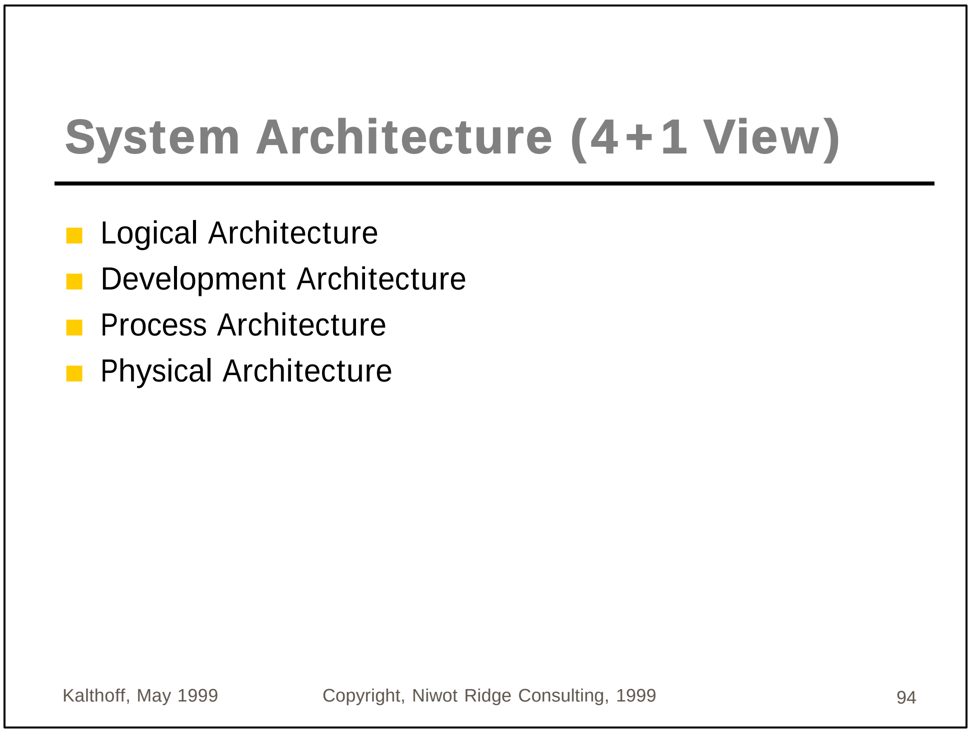


Components of the Architecture

- System Architecture
- Business Architecture
- Operational Architecture

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 93

- Now that we have talked around the theory of architecture, let's look at one practical architecture.
- And remember we now understand the difference between theory and practice, marketecture and architecture.

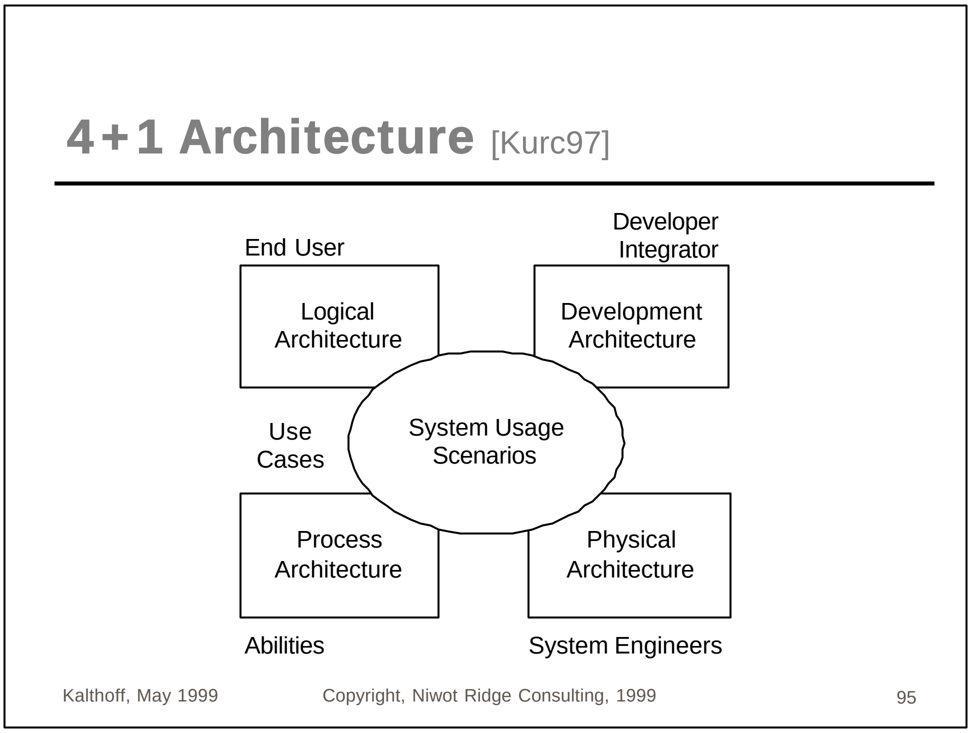


System Architecture (4+1 View)

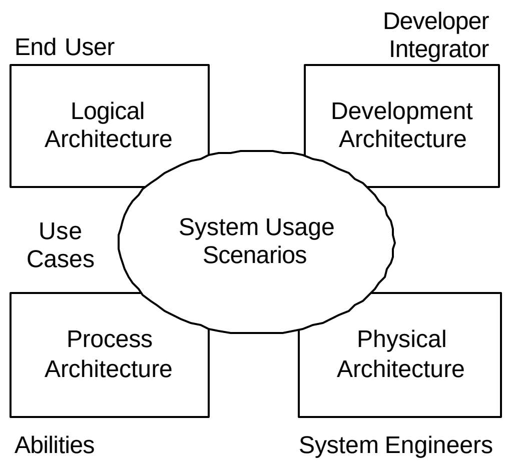
- Logical Architecture
- Development Architecture
- Process Architecture
- Physical Architecture

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 94

- The 4+1 architecture is a very convenient way to decompose systems without creating a bunch of detailed pictures.
- The 4+1 architecture has become the foundation for several other methods and numerous books and articles.

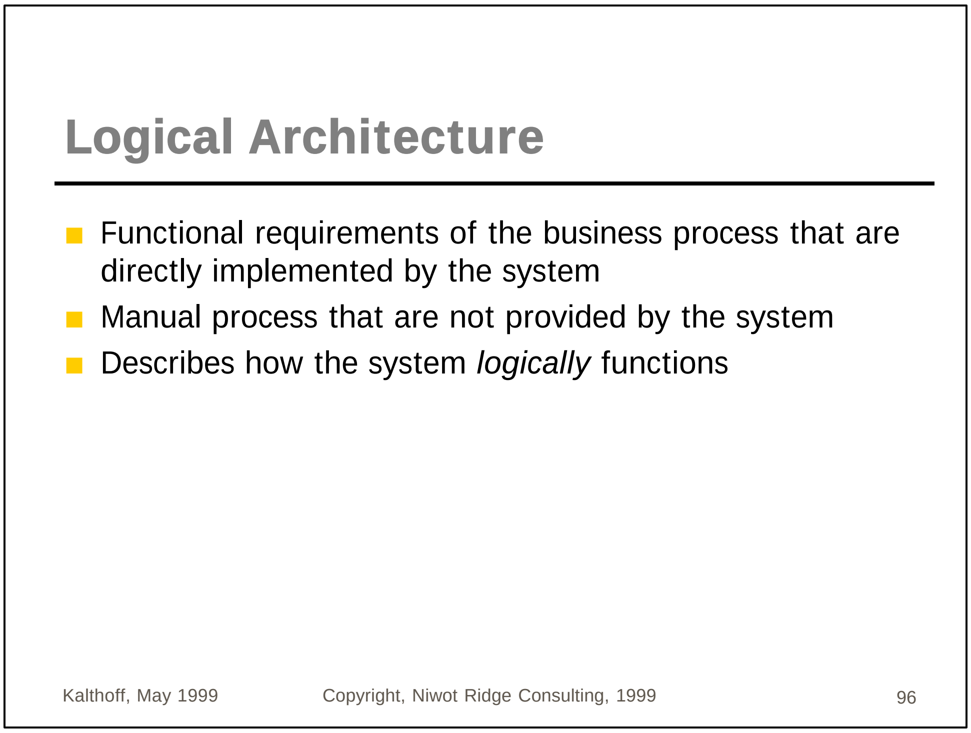


4+1 Architecture [Kurc97]



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 95

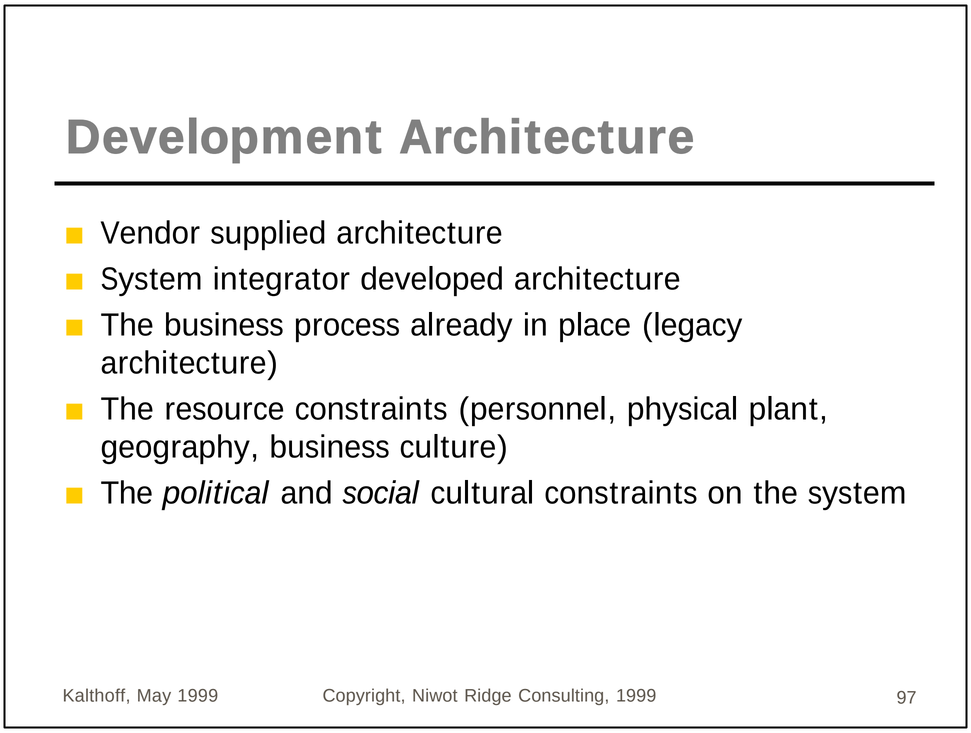
- Here's the simple view of the 4+1 architecture.
- This picture really doesn't convey any software or hardware components...and that's the point.
- It does convey how to discover the software and hardware components, through each *view* of the system.
- Using scenarios the architect can develop the 4+1 views of the system.



Logical Architecture

- Functional requirements of the business process that are directly implemented by the system
- Manual process that are not provided by the system
- Describes how the system *logically* functions

- This is the system seen by the user.
- It consists of external behaviors which react to user stimulus and produce results displayed on the screen or printed on paper.
- For example in an EDMS this architecture would describes the search process for documents:
  - Tree walking.
  - Content based retrieval.
  - Application image enabling.

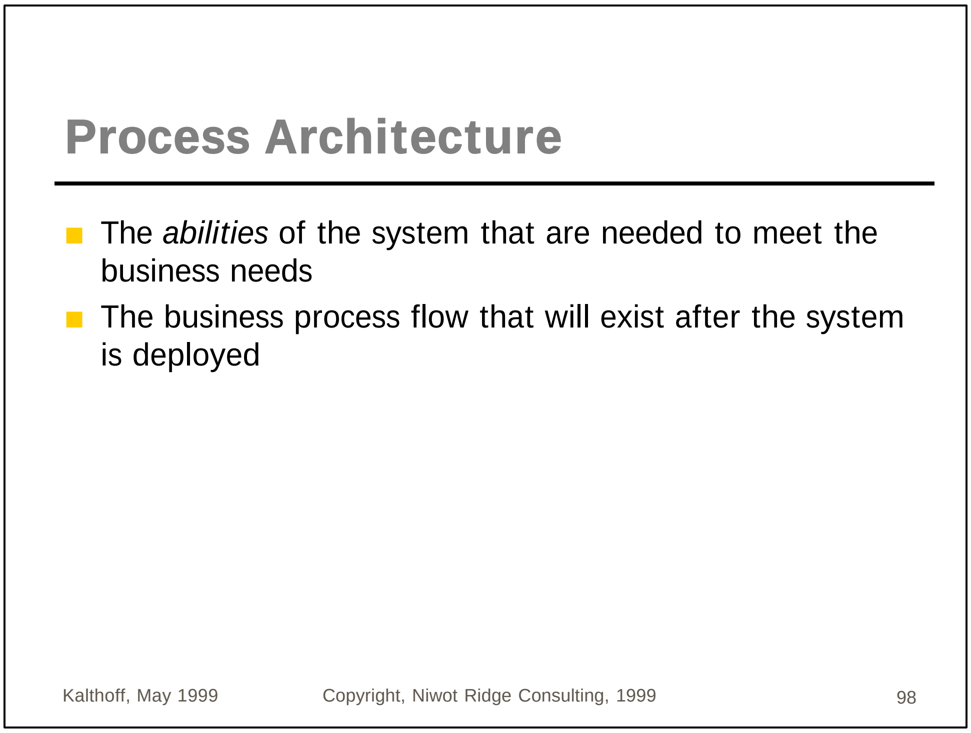


Development Architecture

- Vendor supplied architecture
- System integrator developed architecture
- The business process already in place (legacy architecture)
- The resource constraints (personnel, physical plant, geography, business culture)
- The *political* and *social* cultural constraints on the system

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 97

- This is how the system is organized.

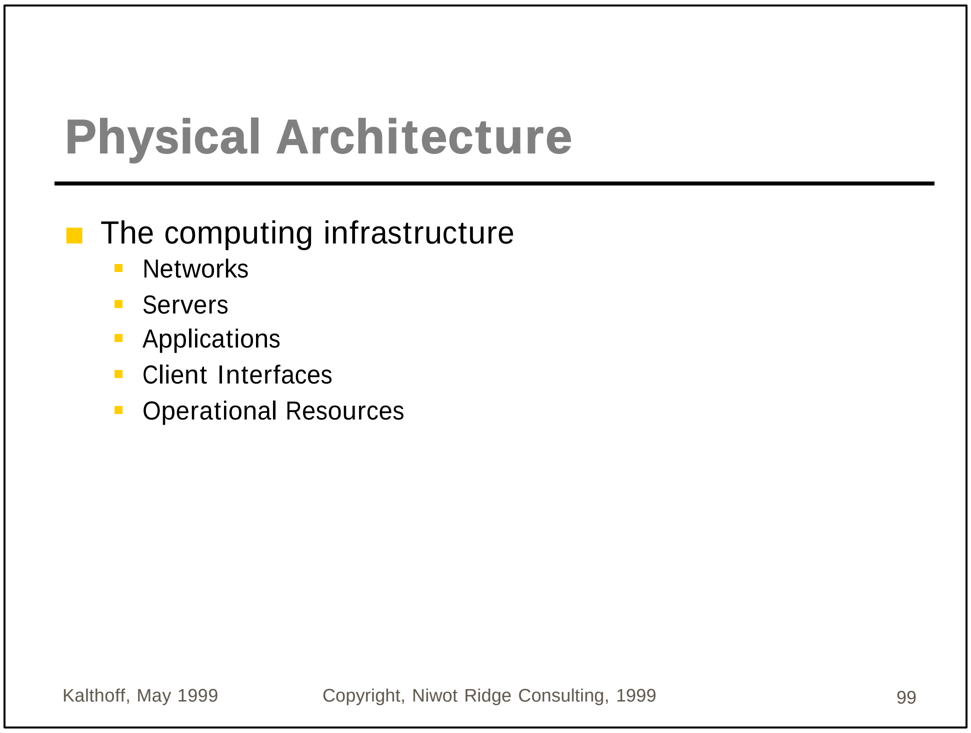


Process Architecture

- The *abilities* of the system that are needed to meet the business needs
- The business process flow that will exist after the system is deployed

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 98

- These are the non–functional architecture components.
- These are also the business process flows that provide the glue between the processing steps.

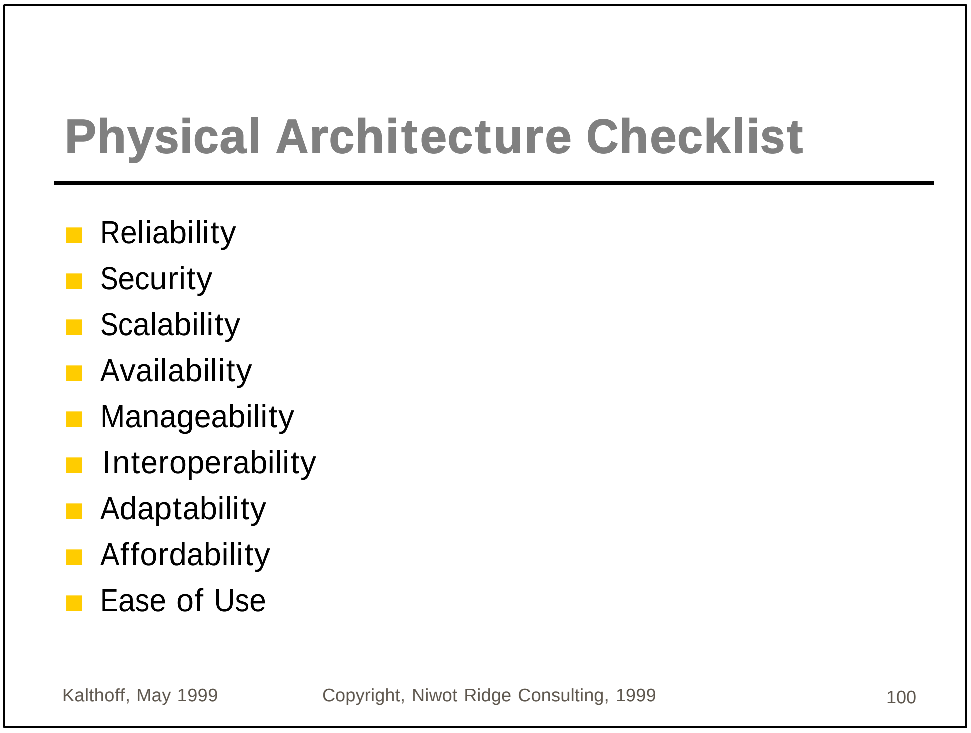


Physical Architecture

- The computing infrastructure
  - Networks
  - Servers
  - Applications
  - Client Interfaces
  - Operational Resources

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 99

- This is the infrastructure view and the most common boxicological description.
- The process view is also a common view since boxes actually do describe business processes.

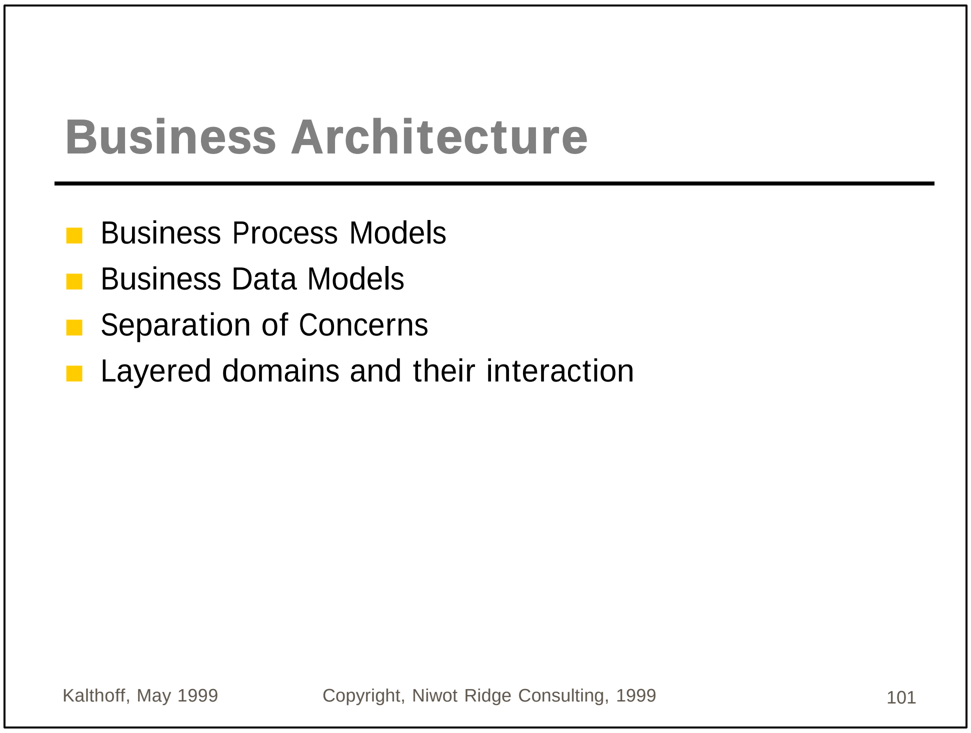


Physical Architecture Checklist

- Reliability
- Security
- Scalability
- Availability
- Manageability
- Interoperability
- Adaptability
- Affordability
- Ease of Use

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 100

- In addition to the 4+1 views there are some derived views of the physical and business architecture.
- Here is a *check list* for the physical architecture components.
- This check list is from Sun and is designed to bring out the differences between Solaris and NT.
- It will still work to bring out the the details of the hardware and networking components of the system.

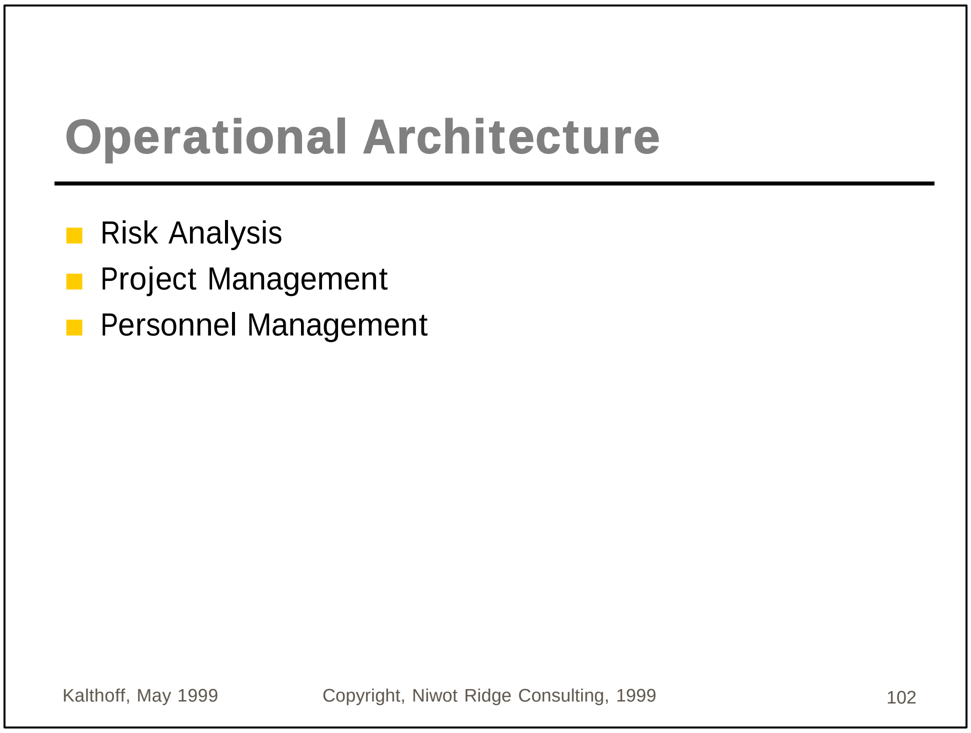


Business Architecture

- Business Process Models
- Business Data Models
- Separation of Concerns
- Layered domains and their interaction

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 101

- The 4+1 view has an end user process, but the details of the business model also needs development.
- These include the business activities that use the system and the behaviors of the users.

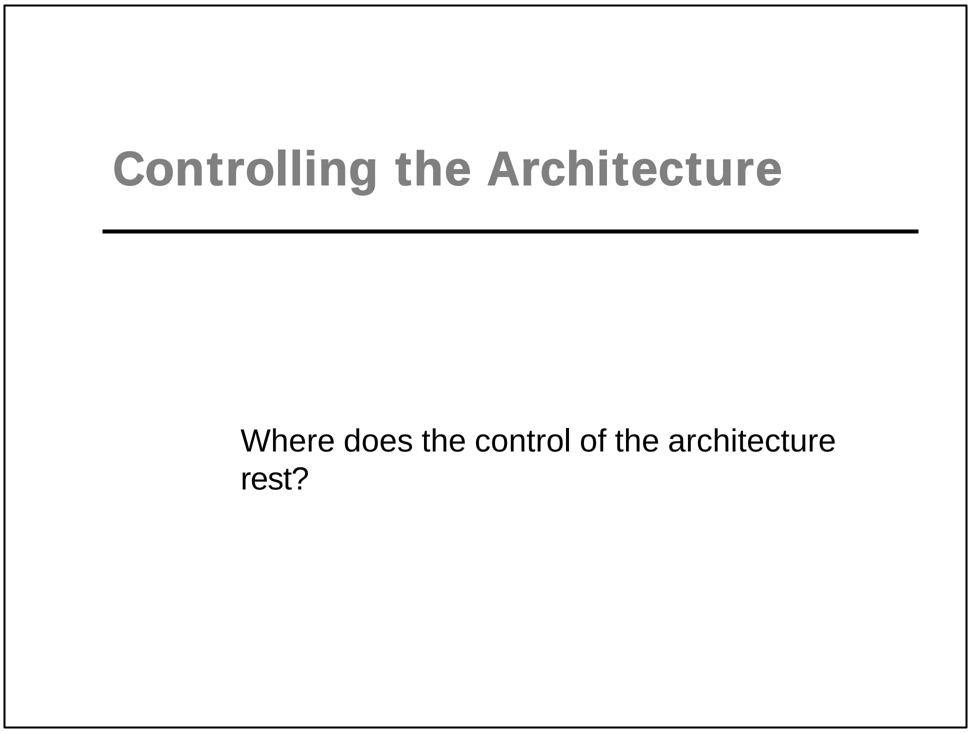


Operational Architecture

- Risk Analysis
- Project Management
- Personnel Management

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 102

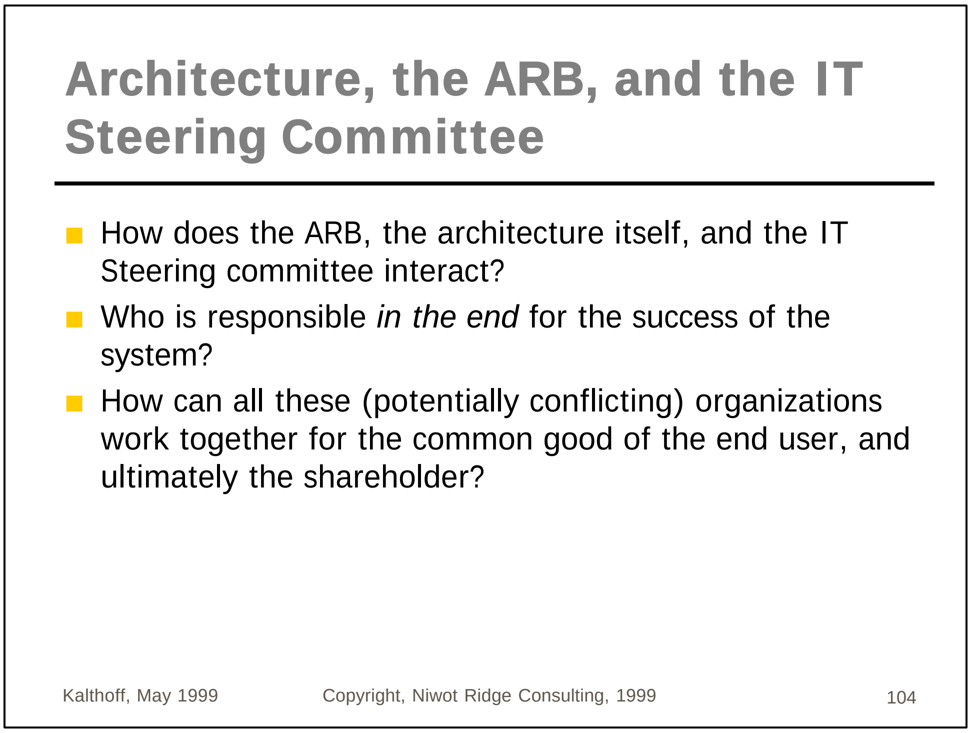
- There are operational aspects of the system that are not covered by the 4+1 architecture.
- These are part of good business practices, but are overlooked.
- The risk analysis process is a Critical Success Factor for the project and should not be overlooked.
- There are software programs for managing risk and several good books for defining how to set up and manage risk management project.
- The project manager and the architect should be responsible for the risk management.



Controlling the Architecture

Where does the control of the architecture rest?

- Now we get to the point...how does all this information get deployed and managed?

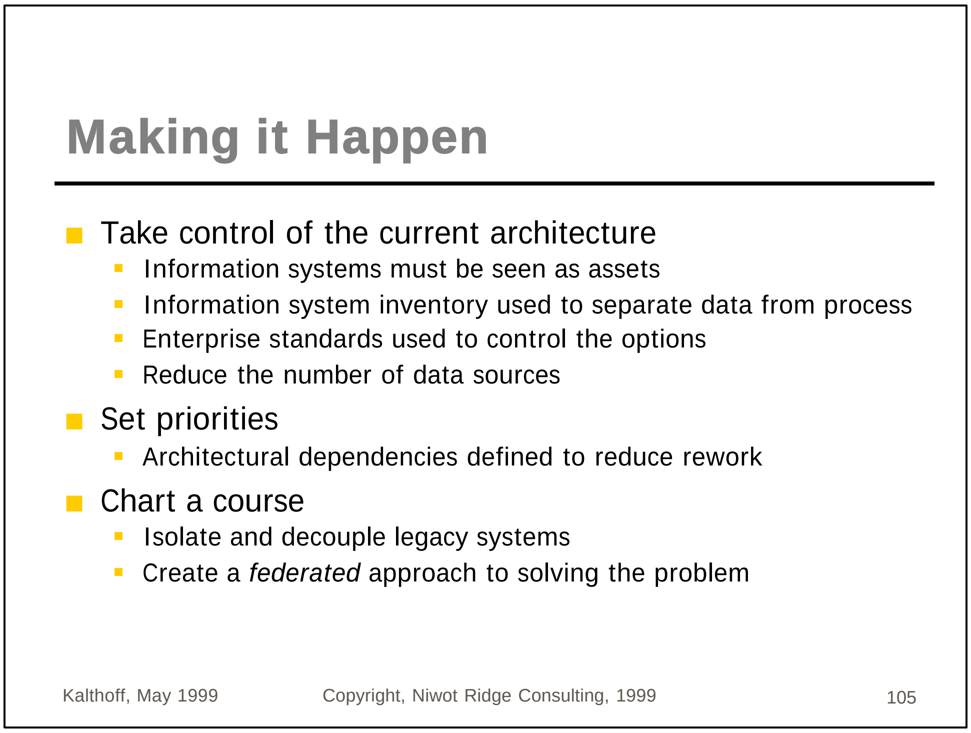


Architecture, the ARB, and the IT Steering Committee

- How does the ARB, the architecture itself, and the IT Steering committee interact?
- Who is responsible *in the end* for the success of the system?
- How can all these (potentially conflicting) organizations work together for the common good of the end user, and ultimately the shareholder?

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 104

- The title of this session, at least when the proceedings were printed had Architectural Review Board in it.
- So, let’s assume that the ARB will be part of the solution.
- The real solution can be called anything you want, it’s the behaviors we’re after.
- But there are potential conflicts between the IT Steering committee, the architect, and the ARB, as well as the project manager.
- Working these out is a major effort and a critical success factor.

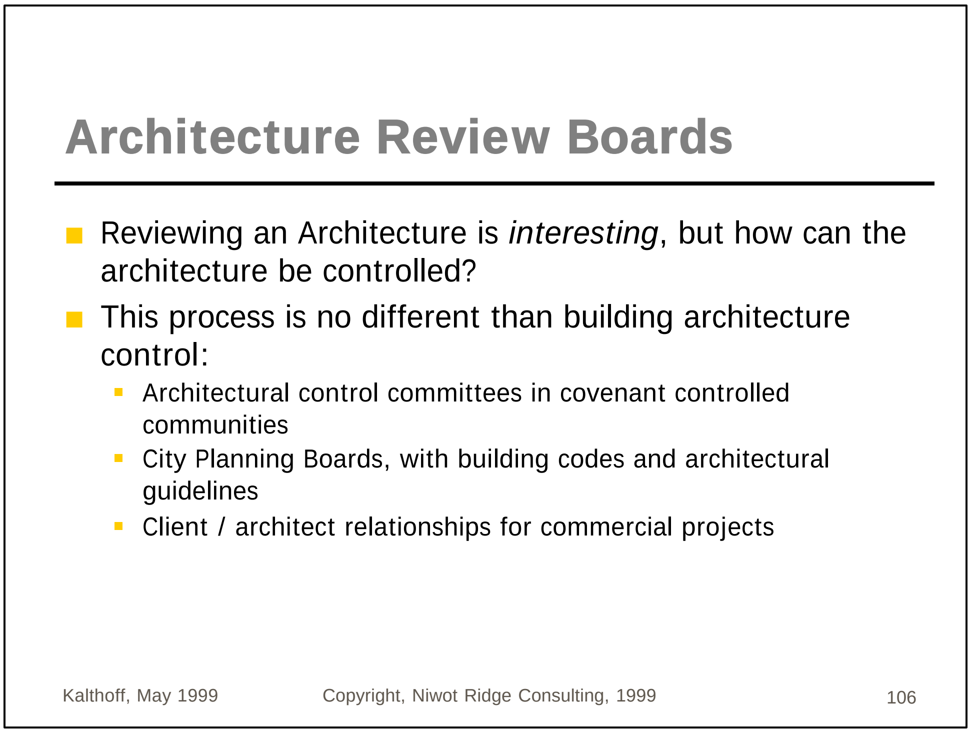


Making it Happen

- Take control of the current architecture
  - Information systems must be seen as assets
  - Information system inventory used to separate data from process
  - Enterprise standards used to control the options
  - Reduce the number of data sources
- Set priorities
  - Architectural dependencies defined to reduce rework
- Chart a course
  - Isolate and decouple legacy systems
  - Create a *federated* approach to solving the problem

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 105

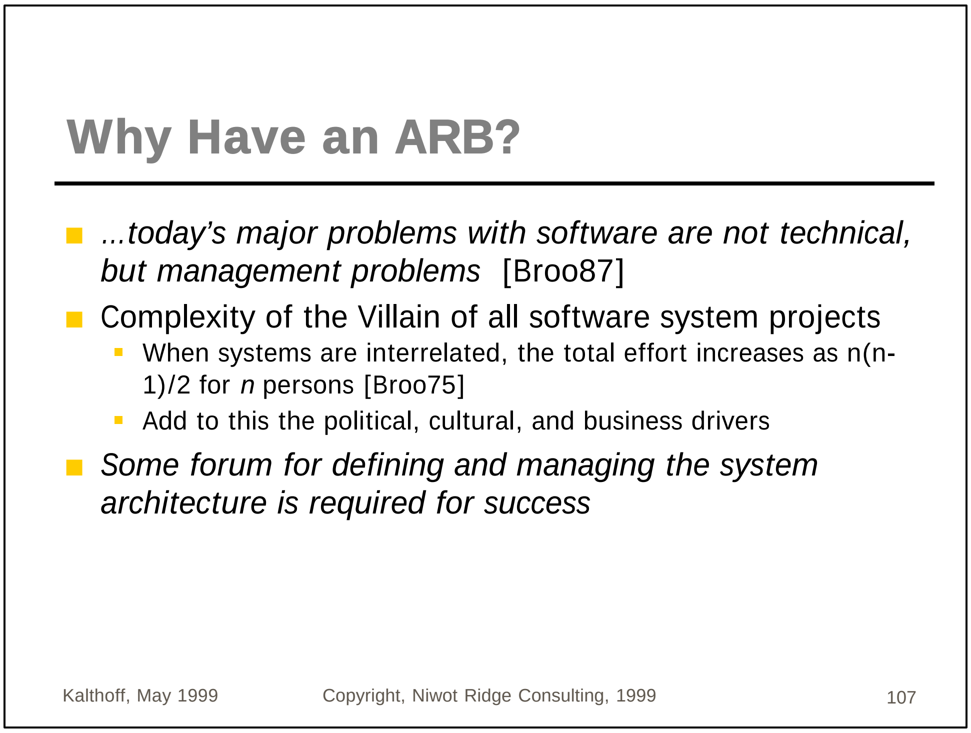
- If you were assigned to lead the ARB next week - after attending this seminar - here’s a starting set of goals and objectives.



Architecture Review Boards

- Reviewing an Architecture is *interesting*, but how can the architecture be controlled?
- This process is no different than building architecture control:
  - Architectural control committees in covenant controlled communities
  - City Planning Boards, with building codes and architectural guidelines
  - Client / architect relationships for commercial projects

- In the ARB we'll assume the word review is a verb.

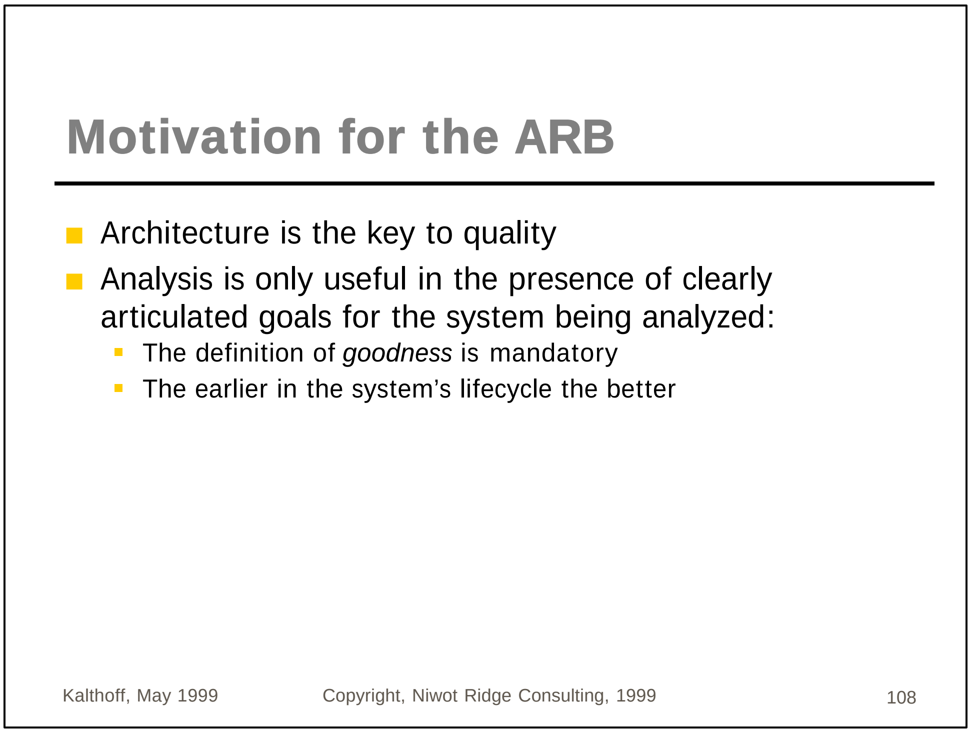


Why Have an ARB?

- *...today's major problems with software are not technical, but management problems* [Broo87]
- Complexity of the Villain of all software system projects
  - When systems are interrelated, the total effort increases as n(n-1)/2 for *n* persons [Broo75]
  - Add to this the political, cultural, and business drivers
- *Some forum for defining and managing the system architecture is required for success*

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 107

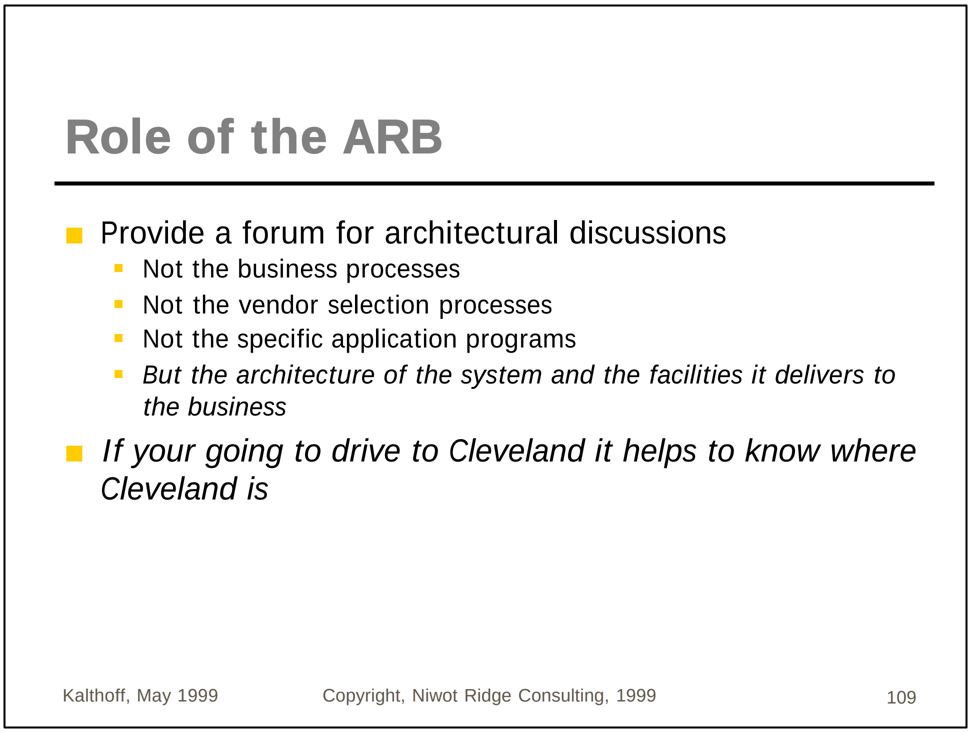
- Just to remind ourselves, the problems which the ARB is trying to address are the non-technical as well as technical aspects of the system.


Motivation for the ARB

- Architecture is the key to quality
- Analysis is only useful in the presence of clearly articulated goals for the system being analyzed:
  - The definition of *goodness* is mandatory
  - The earlier in the system's lifecycle the better

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 108

- If we follow the lead of the the session title, the ARB is based on good architectural practices.
- Using Chris Alexander's writing, quality is the essence of architecture.
- This is not quality in the sense of 6 sigma statistical control quality. This is the quality that makes good architecture stand out from bad architecture.

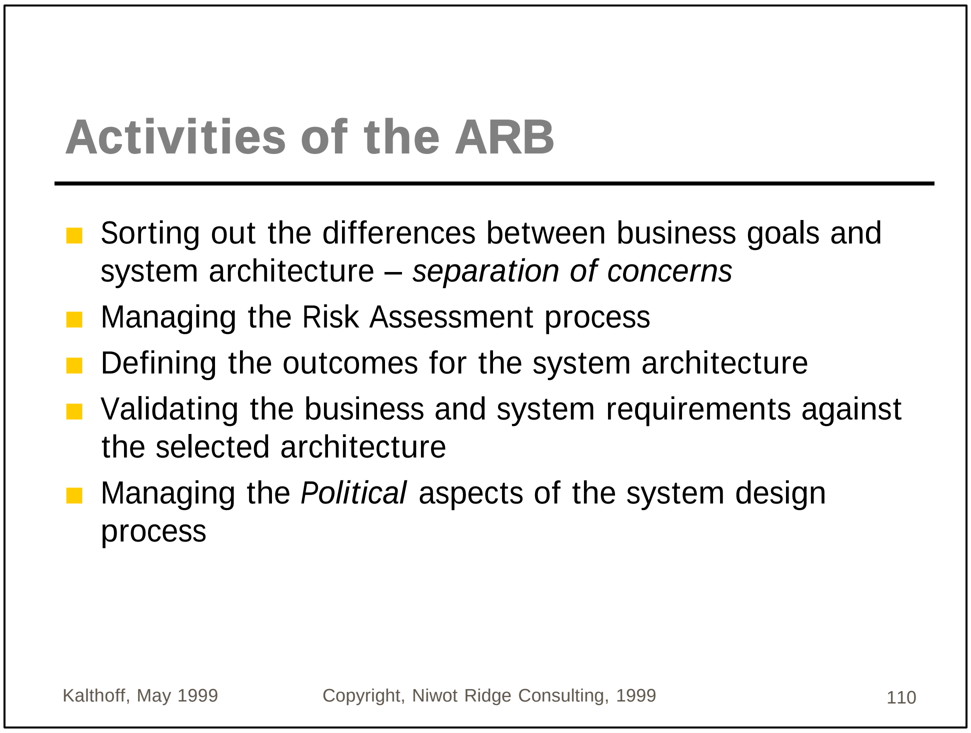


Role of the ARB

- Provide a forum for architectural discussions
  - Not the business processes
  - Not the vendor selection processes
  - Not the specific application programs
  - *But the architecture of the system and the facilities it delivers to the business*
- *If your going to drive to Cleveland it helps to know where Cleveland is*

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 109

- Before any ARB can define the successful outcomes, there must be be some discussion of what the ARB will not be responsible for.

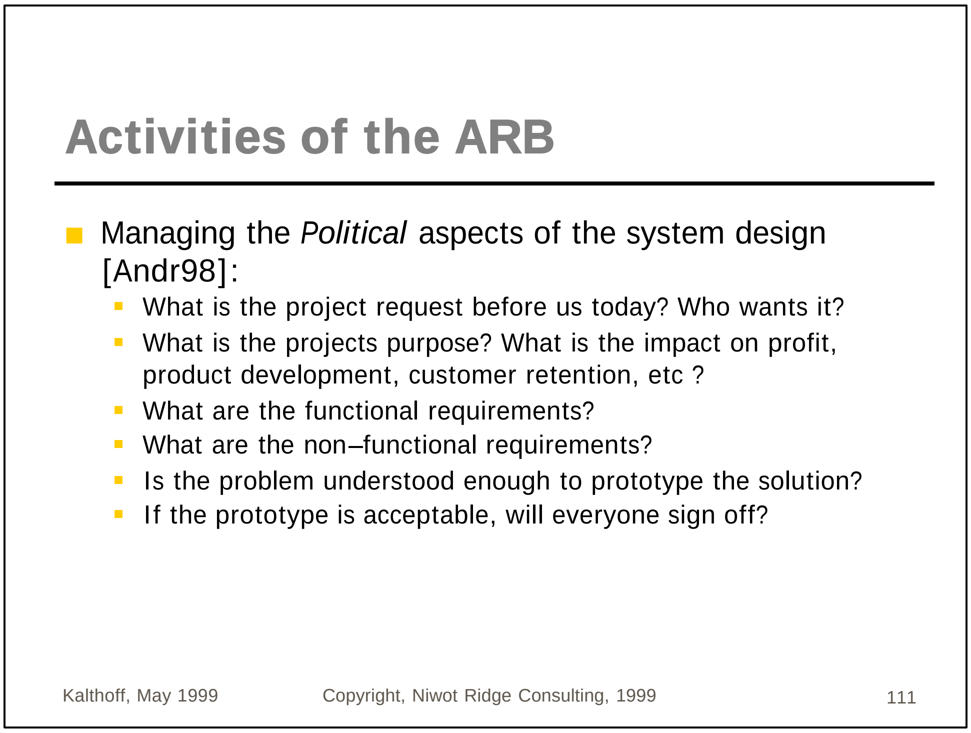


Activities of the ARB

- Sorting out the differences between business goals and system architecture – *separation of concerns*
- Managing the Risk Assessment process
- Defining the outcomes for the system architecture
- Validating the business and system requirements against the selected architecture
- Managing the *Political* aspects of the system design process

Kalthoff, May 1999 110

- Here are some activities the ARB should perform during its life.

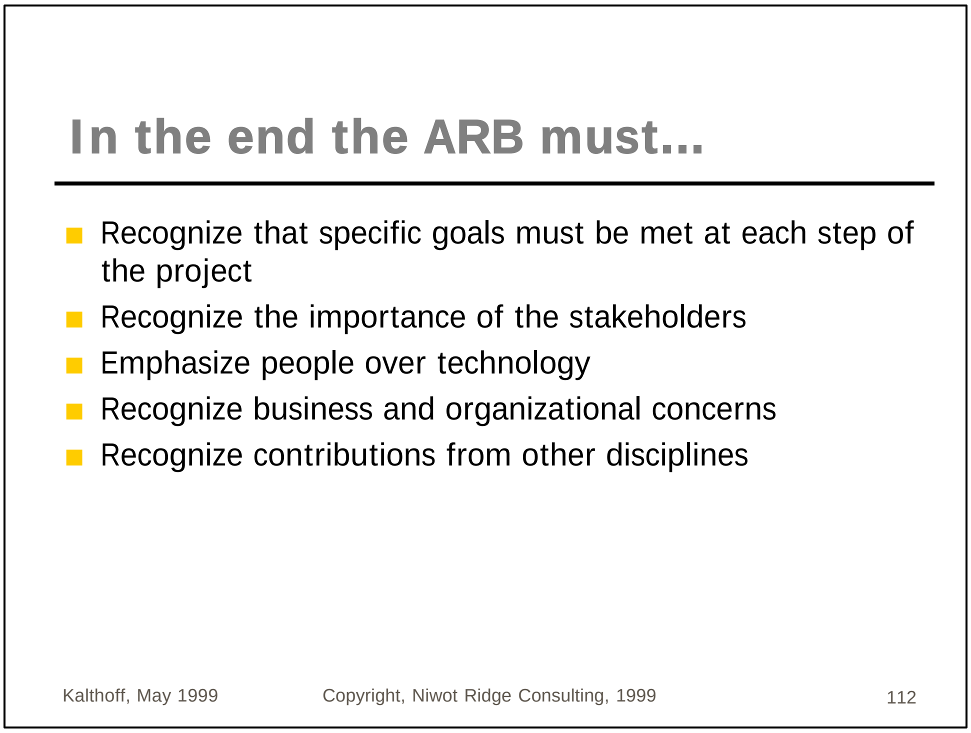


Activities of the ARB

- Managing the *Political* aspects of the system design [Andr98]:
  - What is the project request before us today? Who wants it?
  - What is the projects purpose? What is the impact on profit, product development, customer retention, etc ?
  - What are the functional requirements?
  - What are the non–functional requirements?
  - Is the problem understood enough to prototype the solution?
  - If the prototype is acceptable, will everyone sign off?

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 111

- Besides the technical aspects of the ARB, which will be the primary focus of the working groups, the political aspects will consume much of the members time.
- Why, because humans are involved and money is being spent, and careers are being impacted.
- Add all those forces together and you get politics.

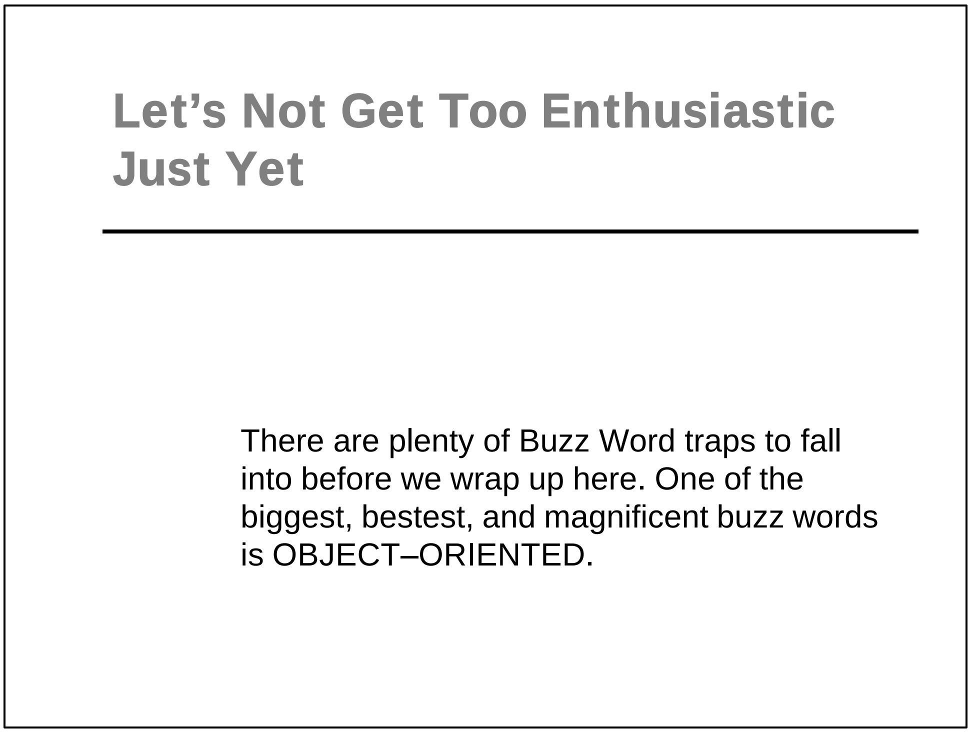


In the end the ARB must...

- Recognize that specific goals must be met at each step of the project
- Recognize the importance of the stakeholders
- Emphasize people over technology
- Recognize business and organizational concerns
- Recognize contributions from other disciplines

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 112

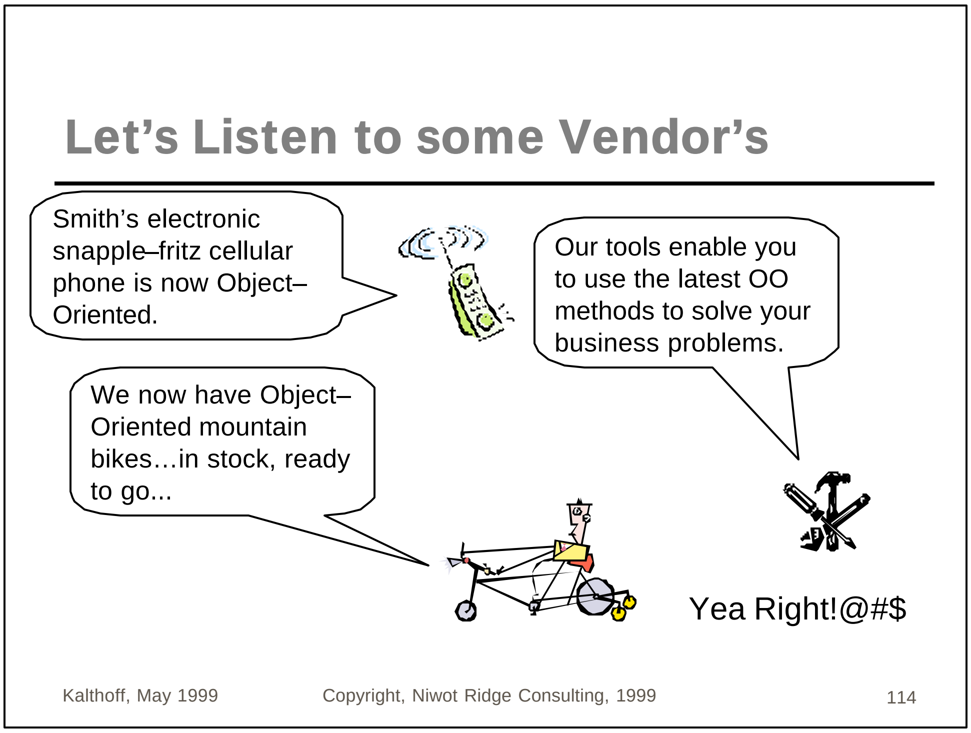
- No matter what technical outcomes result from the board and its subcommittees, a framework for success must be adopted in the beginning.
- Here is one, or at least one that can be a start.



Let’s Not Get Too Enthusiastic Just Yet

There are plenty of Buzz Word traps to fall into before we wrap up here. One of the biggest, bestest, and magnificent buzz words is OBJECT–ORIENTED.

- Before we leave the current subject and move on to the Architecture Review Board, let’s not forget the one overpowering force in all the architecture domain – Objects.

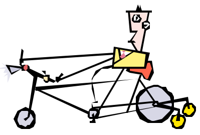


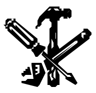
Let’s Listen to some Vendor’s

Smith’s electronic snapple–fritz cellular phone is now Object–Oriented.

Our tools enable you to use the latest OO methods to solve your business problems.

We now have Object–Oriented mountain bikes…in stock, ready to go...

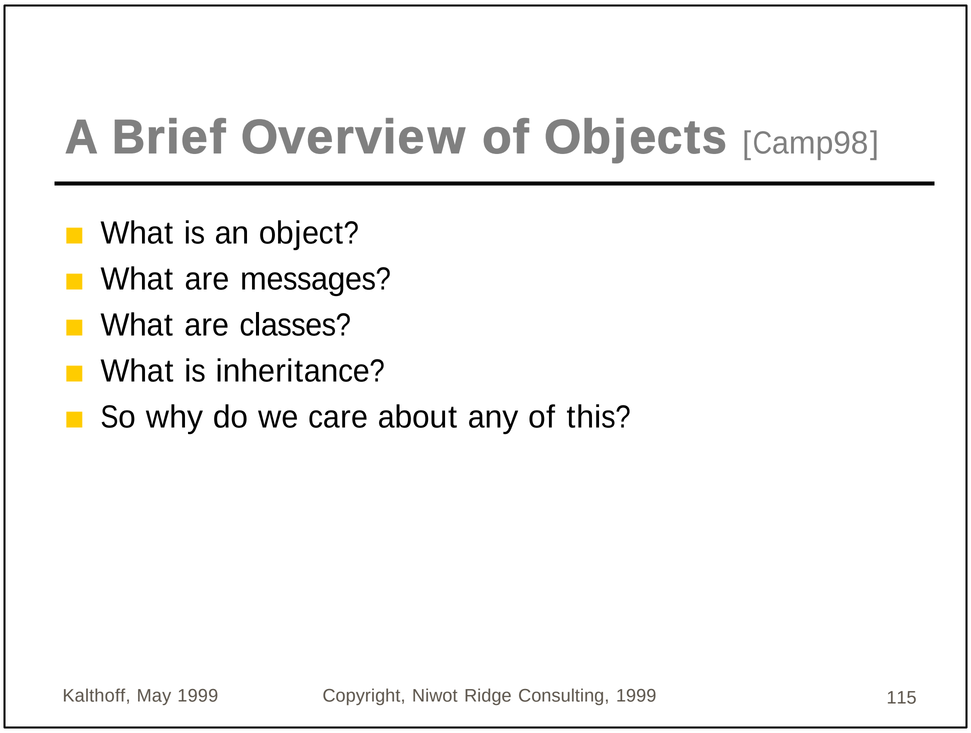




Yea Right!@#$

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 114

- The term object is a wonderful buzz word.
- The perfect buzz word.
- Everyone uses it, and very few people actually know what it means in the technical sense.
- Calling something an object does not make it one.

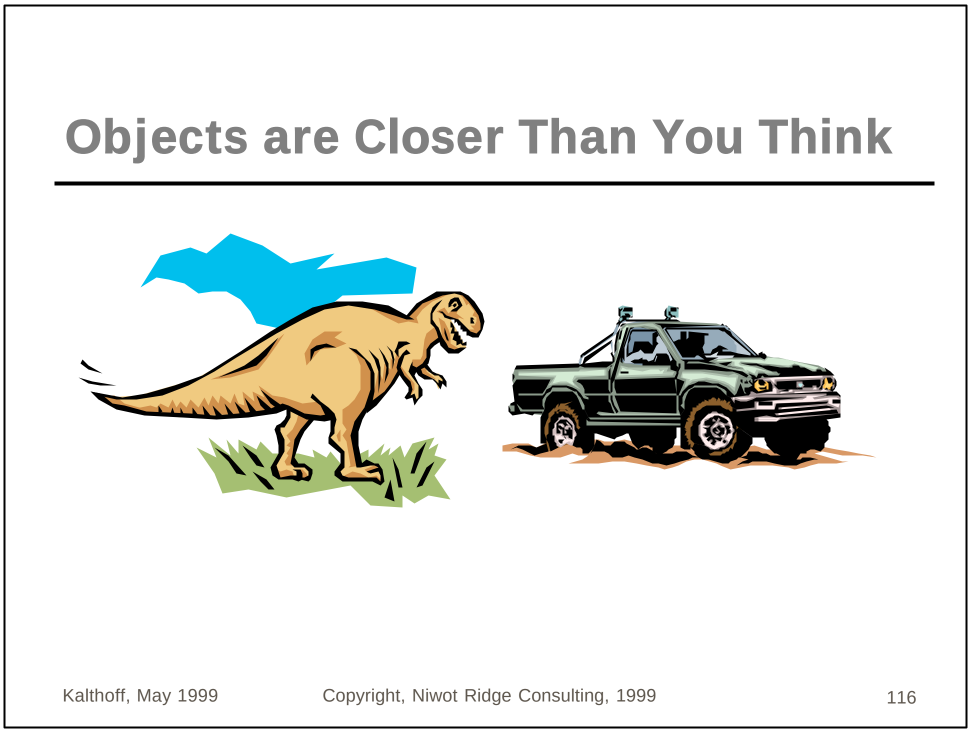


A Brief Overview of Objects [Camp98]

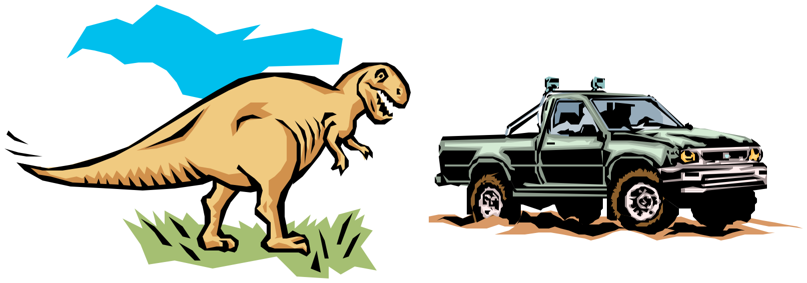
- What is an object?
- What are messages?
- What are classes?
- What is inheritance?
- So why do we care about any of this?

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 115

- Our ground time here will be short, so stretch your legs and lets have a quick look at the term Object.

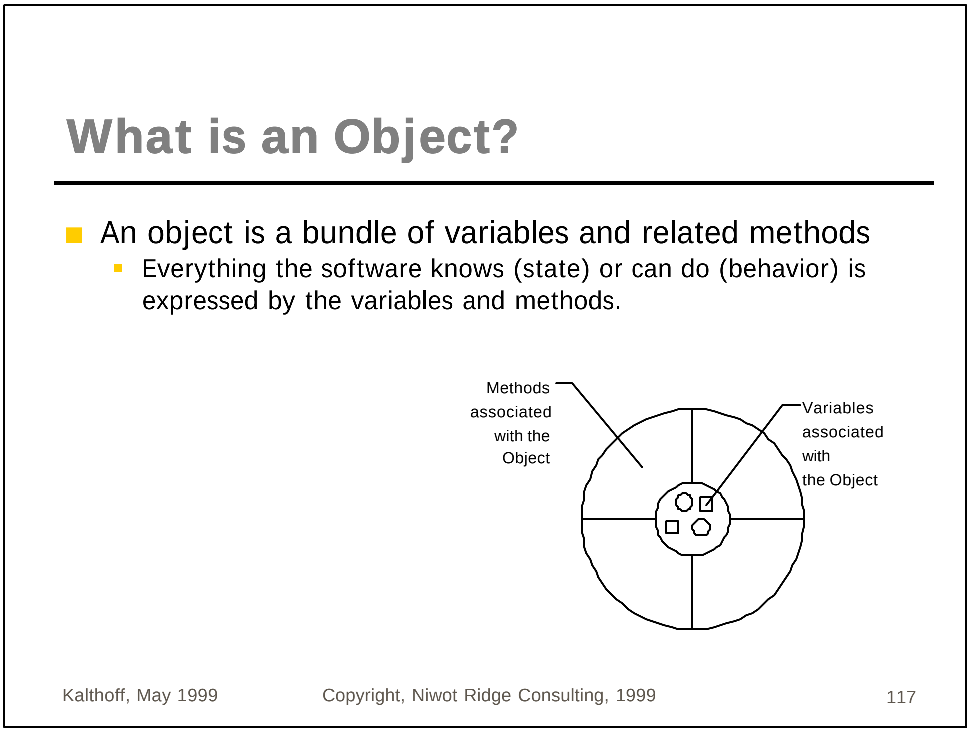


Objects are Closer Than You Think



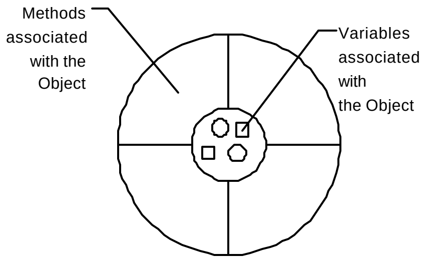
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 116

- In the object world, the use of architectural paradigm comes with the territory.
- Outside the object world, I really mean object programming, which is Java, C++, and Smalltalk, and the object standards, such as CORBA, objects are seen as somehow mystical.



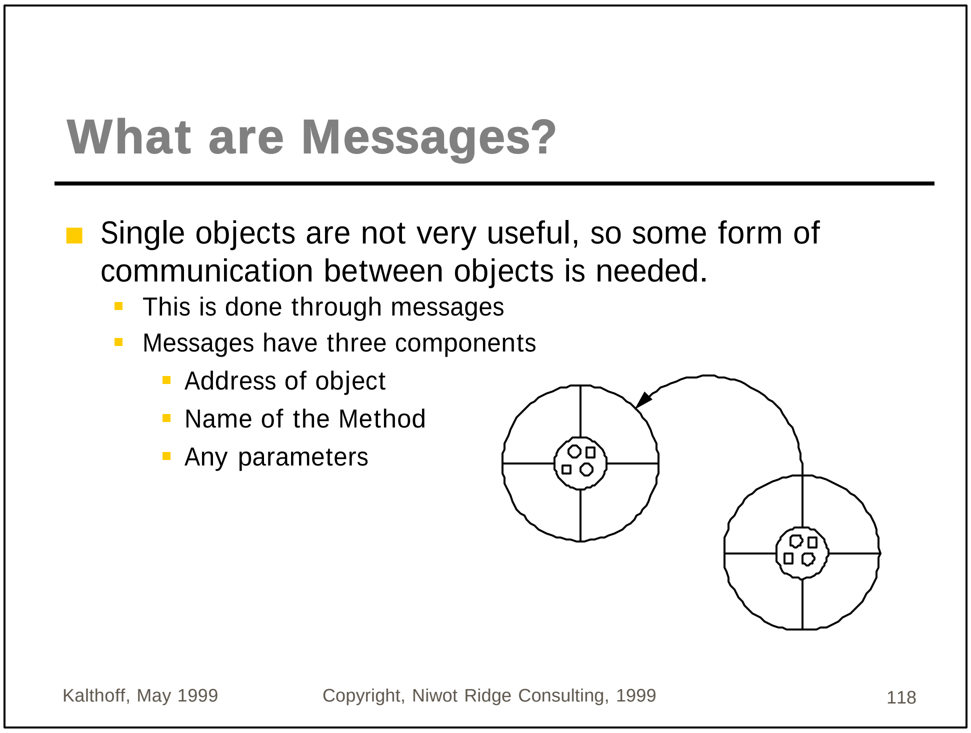
What is an Object?

- An object is a bundle of variables and related methods
  - Everything the software knows (state) or can do (behavior) is expressed by the variables and methods.


Methods associated with the Object
Variables associated with the Object

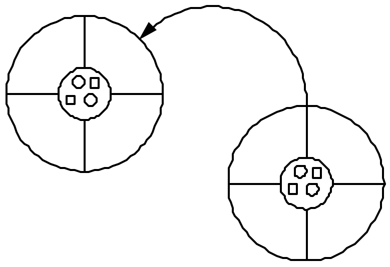
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 117

- An object is a collection of variables and methods.
- Variables describe what states the object can be in and the methods describe which behaviors the object can express.
- If this object is a bicycle the methods would represent:
  - changing gears.
  - Braking.
- The variables represent the current gear and the current speed of the bike.



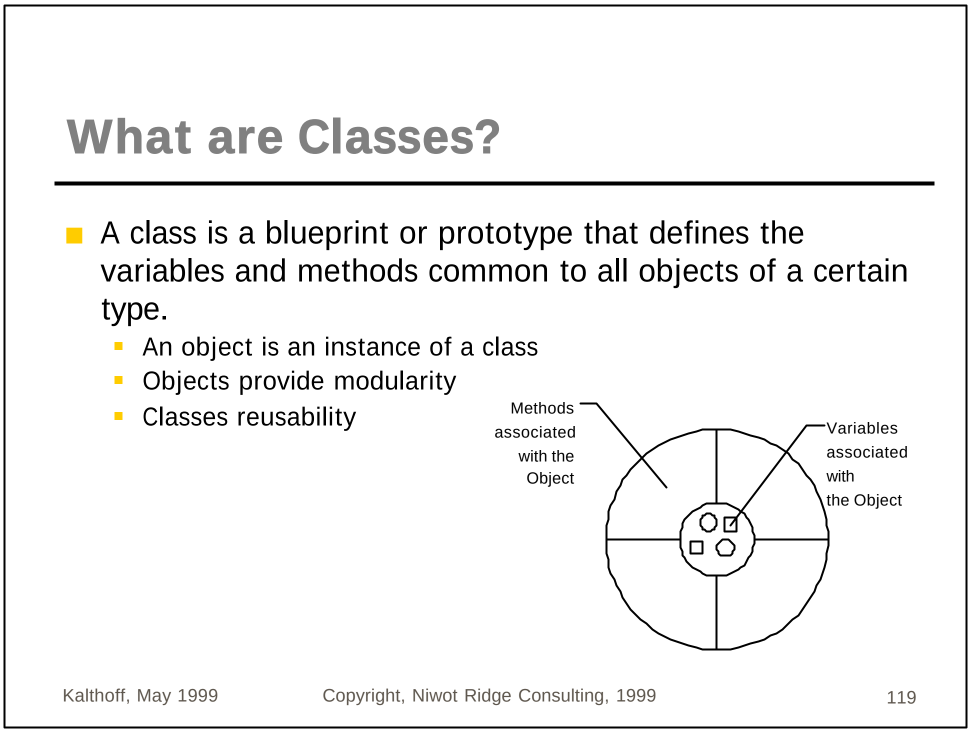
What are Messages?

- Single objects are not very useful, so some form of communication between objects is needed.
  - This is done through messages
  - Messages have three components
    - Address of object
    - Name of the Method
    - Any parameters



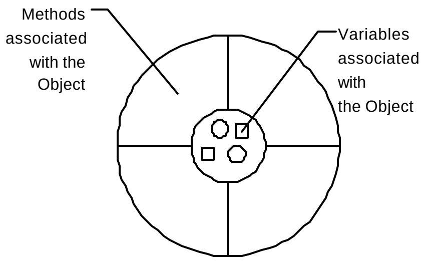
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 118

- In order for objects to interact with other objects a communication mechanism is needed. In the majority of object languages this is done through messages.
- If our objects here represented a bicycle and a rider, the rider may send the bicycle object a message to change to a lower gear.



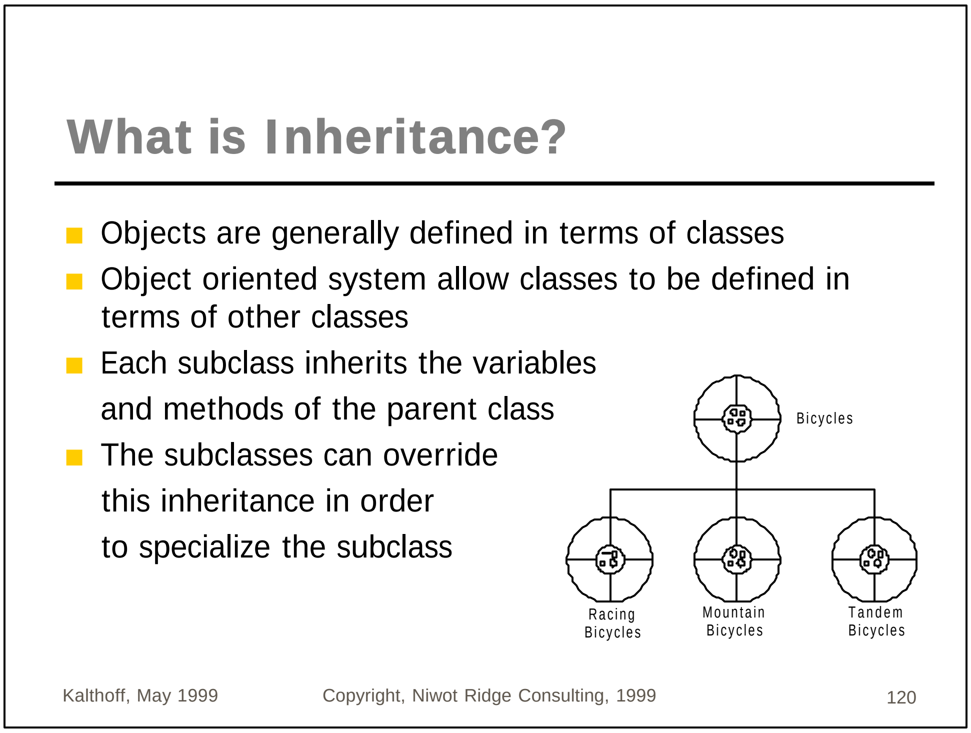
What are Classes?

- A class is a blueprint or prototype that defines the variables and methods common to all objects of a certain type.
  - An object is an instance of a class
  - Objects provide modularity
  - Classes reusability


Methods associated with the Object
Variables associated with the Object

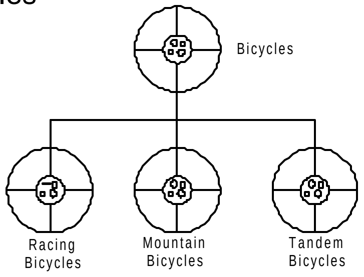
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 119

- There is much confusion between Object, Class and instance.
- This area of objects and their representation goes back to Greek Philosophy, what is a thing and how do we name it.
- In the real world classes are not themselves the objects they describe. There is a real bicycle, a class of bicycles and an instance of a specific class of bicycle.
- In the software world this may be less formal. And the term *object* gets applied inconsistently. It is many times used to refer to both classes and instances of classes.



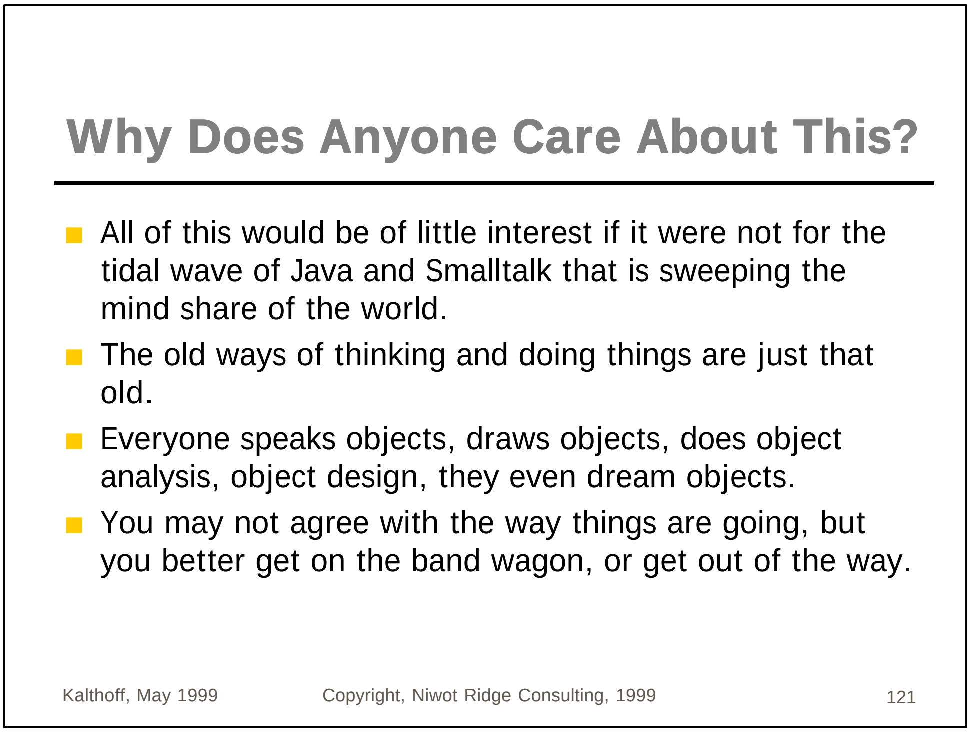
What is Inheritance?

- Objects are generally defined in terms of classes
- Object oriented system allow classes to be defined in terms of other classes
- Each subclass inherits the variables and methods of the parent class
- The subclasses can override this inheritance in order to specialize the subclass



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 120

- Inheritance is one of the object oriented words that instantly creates confusion, along with polymorphism.
- For inheritance the concept is pretty simple. A class can inherit its variables and methods from super-classes.
- The classes that inherit from the super-class can make local modifications to the variables and methods for the new sub-classed class.
- If an object makes changes to the behaviors of its methods, this is called polymorphism. The object looks the same from the outside, but has different behaviors on the inside.
- Does anyone remember the movie The Return of Martin Gere?



Why Does Anyone Care About This?

- All of this would be of little interest if it were not for the tidal wave of Java and Smalltalk that is sweeping the mind share of the world.
- The old ways of thinking and doing things are just that old.
- Everyone speaks objects, draws objects, does object analysis, object design, they even dream objects.
- You may not agree with the way things are going, but you better get on the band wagon, or get out of the way.

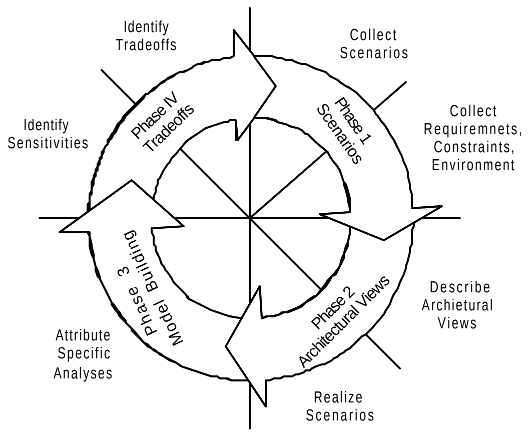
Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 121

- All of this object talk is of little value if we are not in the object world, right? Wrong.
- But the problem is we cannot avoid the object world. No matter if we are programming in an object language or not, the business processes and the data associated with these processes are objects, like it or not.
- So why not adopt a notation and a way of speaking that matches the real world - objects.

After That Brief Object Diversion

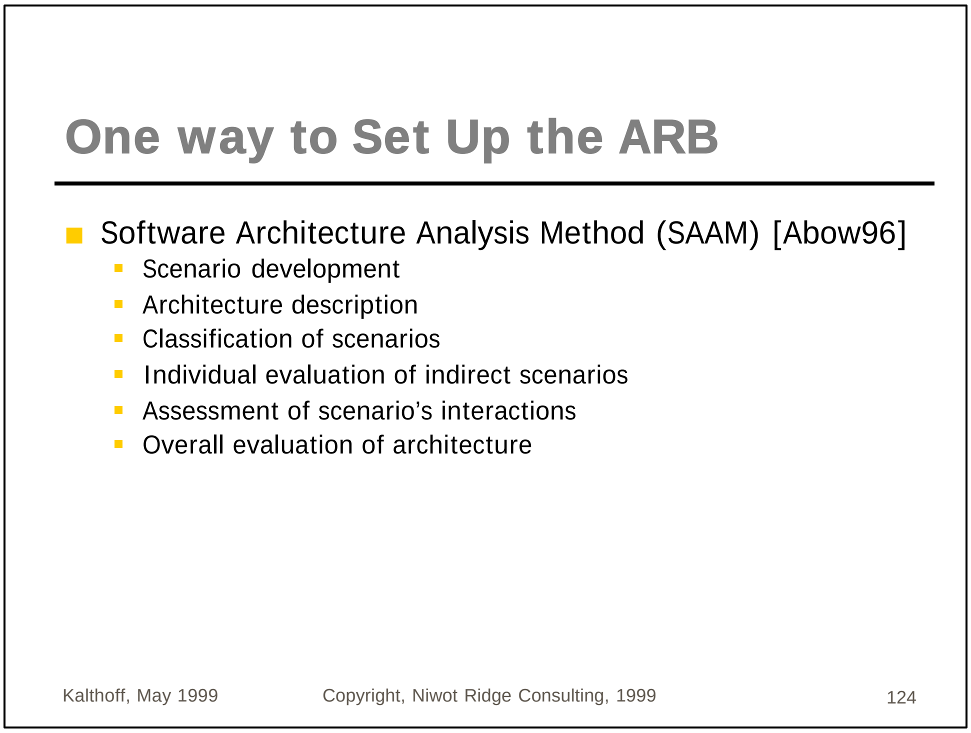
Let's get back to the problem at hand, organizing the architecture and the architectural review board.

Putting It All Together [Kazm98]



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 123

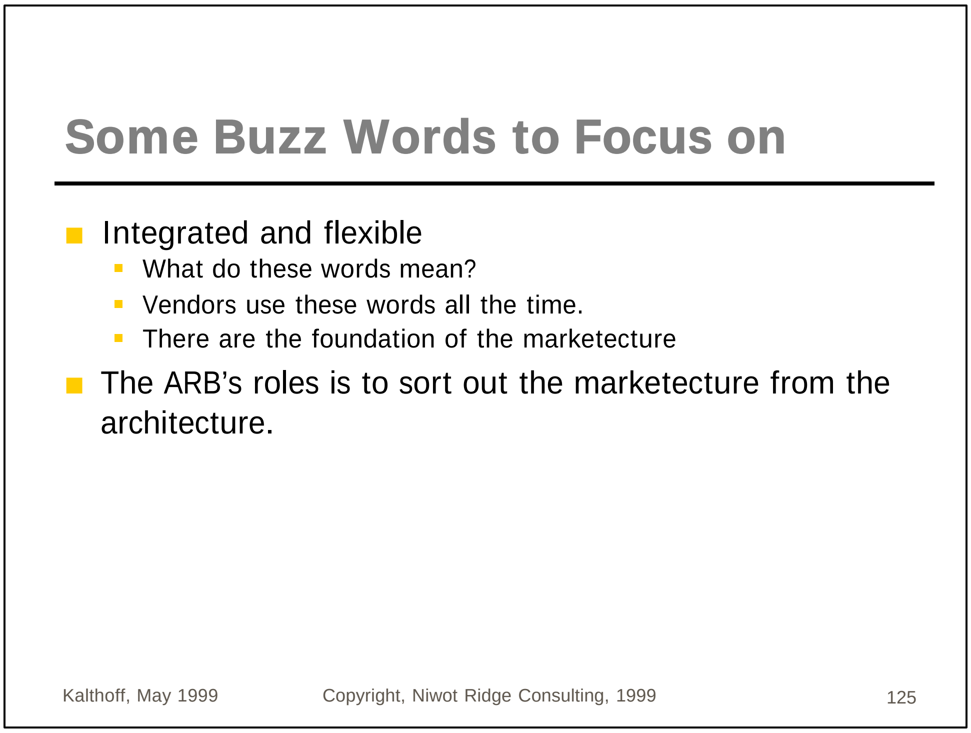
- This picture is taken from the SEI references and is the framework for architectural review.
- You should look through the references to get the details behind the simple idea.
- The actual deployment of these ideas requires hard work and patience, but the payoffs are big.


One way to Set Up the ARB

- Software Architecture Analysis Method (SAAM) [Abow96]
  - Scenario development
  - Architecture description
  - Classification of scenarios
  - Individual evaluation of indirect scenarios
  - Assessment of scenario's interactions
  - Overall evaluation of architecture

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 124

- Using the picture in the previous slide and the methods described in the references, the ARB can start with these processes.
- The key here is to build scenarios which describe how the system will be used and operated.
- These scenarios will be used to ask questions about the architecture.
- A good EDM system question is *how will the system be backed up while it is being used*?
  - Is there on-line backup?
  - Does the system have to be taken down?
  - Can the indexing and document vaults be backed up together as a single entity?
  - What happens if a user wants a single document to be restored?

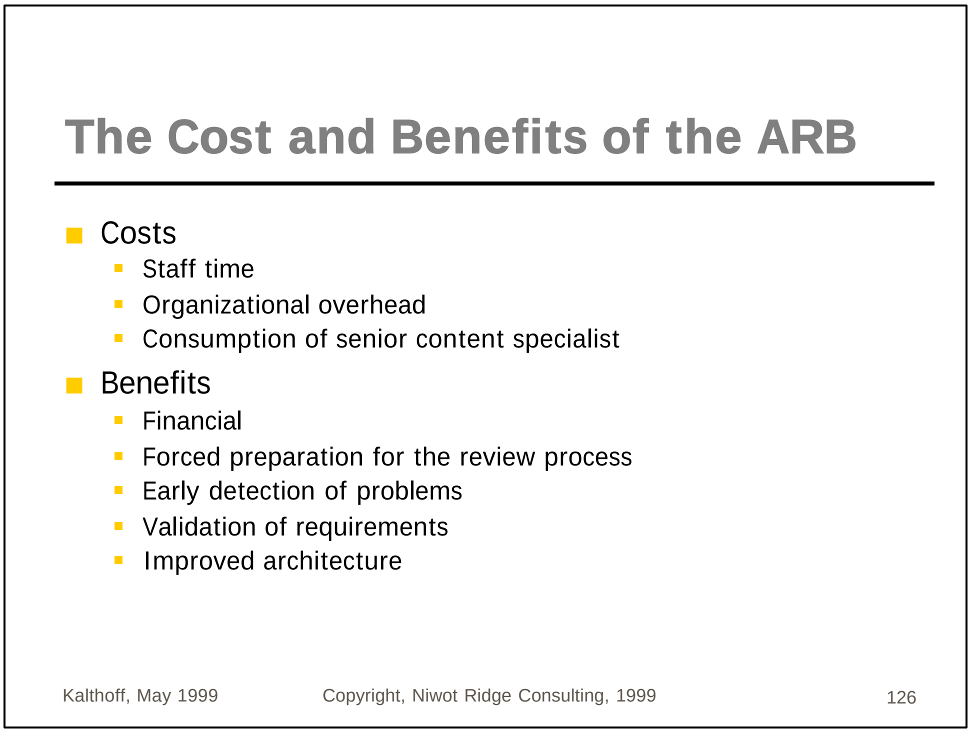


Some Buzz Words to Focus on

- Integrated and flexible
  - What do these words mean?
  - Vendors use these words all the time.
  - There are the foundation of the marketecture
- The ARB's roles is to sort out the marketecture from the architecture.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 125

- The current marketecture requires that words such as integrated and flexible be used during any conversation about a product.
- One of roles of the ARB is to sort out this mess and come up with words that can be measured.
- Integrated? Really, show me how I can attach this application data stream to this processing step of your product?
- Flexible? I want to change the indexing scheme for this class of documents. How can I do that for the next two weeks, than change it back?

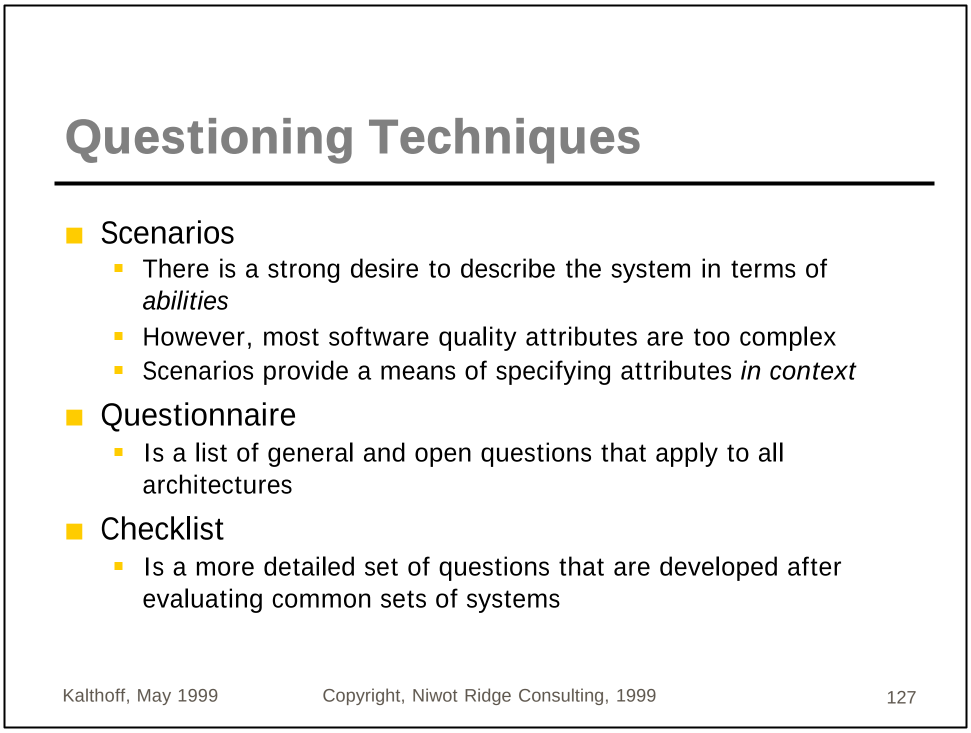


The Cost and Benefits of the ARB

- Costs
  - Staff time
  - Organizational overhead
  - Consumption of senior content specialist
- Benefits
  - Financial
  - Forced preparation for the review process
  - Early detection of problems
  - Validation of requirements
  - Improved architecture

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 126

- The finance folks are always waiting in the wings.
- Here are some bullets they can chew on.

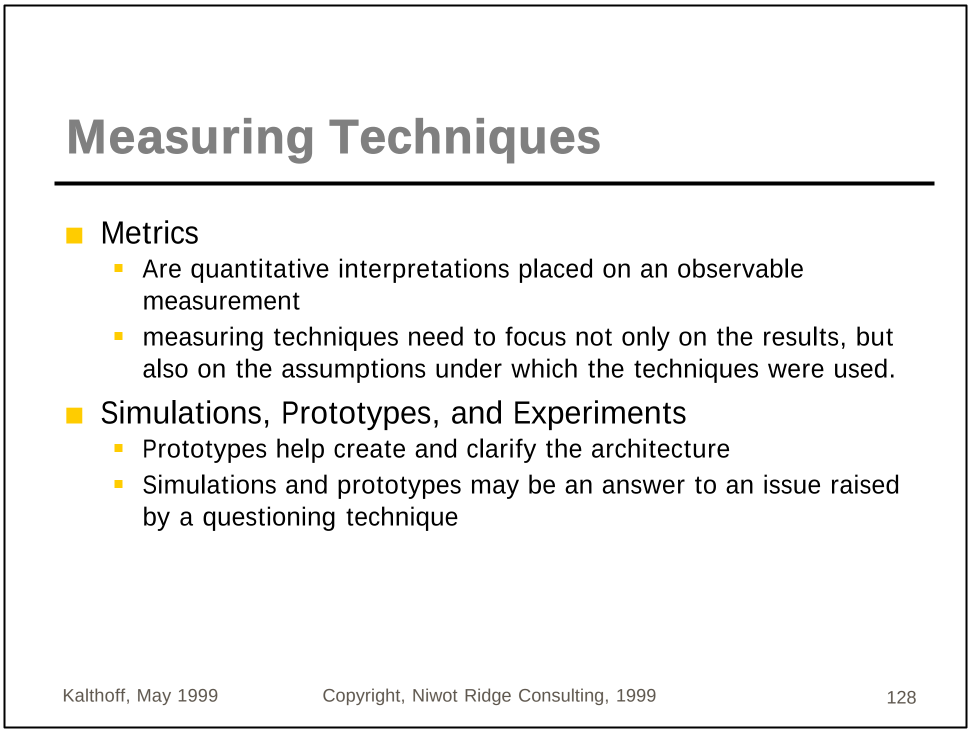


Questioning Techniques

- Scenarios
  - There is a strong desire to describe the system in terms of *abilities*
  - However, most software quality attributes are too complex
  - Scenarios provide a means of specifying attributes *in context*
- Questionnaire
  - Is a list of general and open questions that apply to all architectures
- Checklist
  - Is a more detailed set of questions that are developed after evaluating common sets of systems

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 127

- There must be some structure to the ARB’s architecture control process.
- There are three levels of maturity here. The checklist approach is the most mature, while the scenario approach could be used for a one off architecture review.

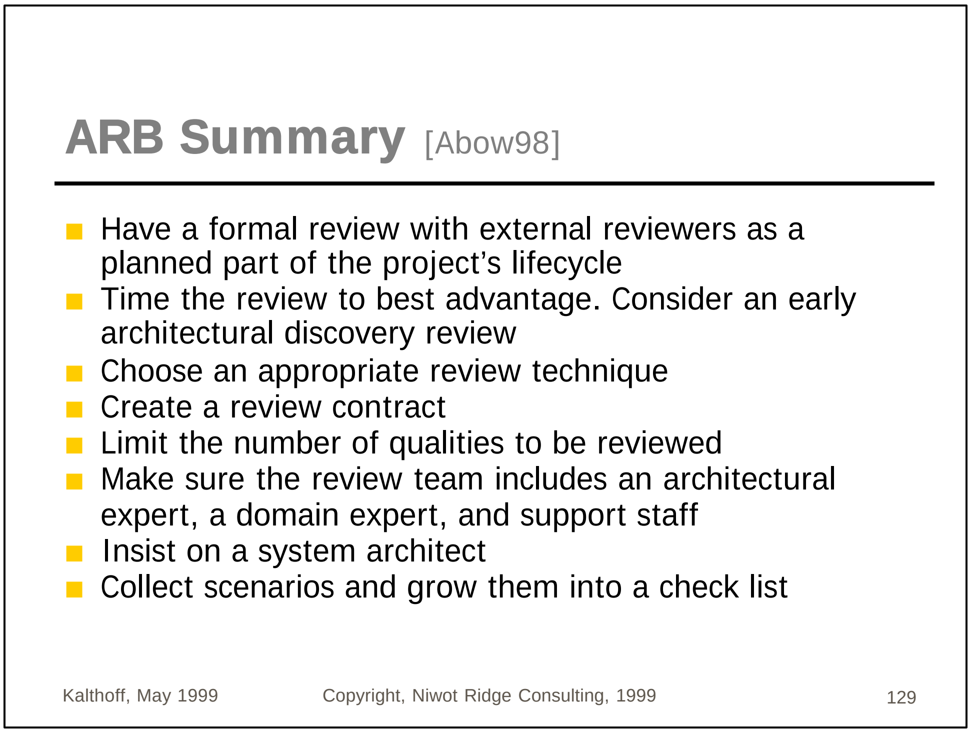


Measuring Techniques

- Metrics
  - Are quantitative interpretations placed on an observable measurement
  - measuring techniques need to focus not only on the results, but also on the assumptions under which the techniques were used.
- Simulations, Prototypes, and Experiments
  - Prototypes help create and clarify the architecture
  - Simulations and prototypes may be an answer to an issue raised by a questioning technique

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 128

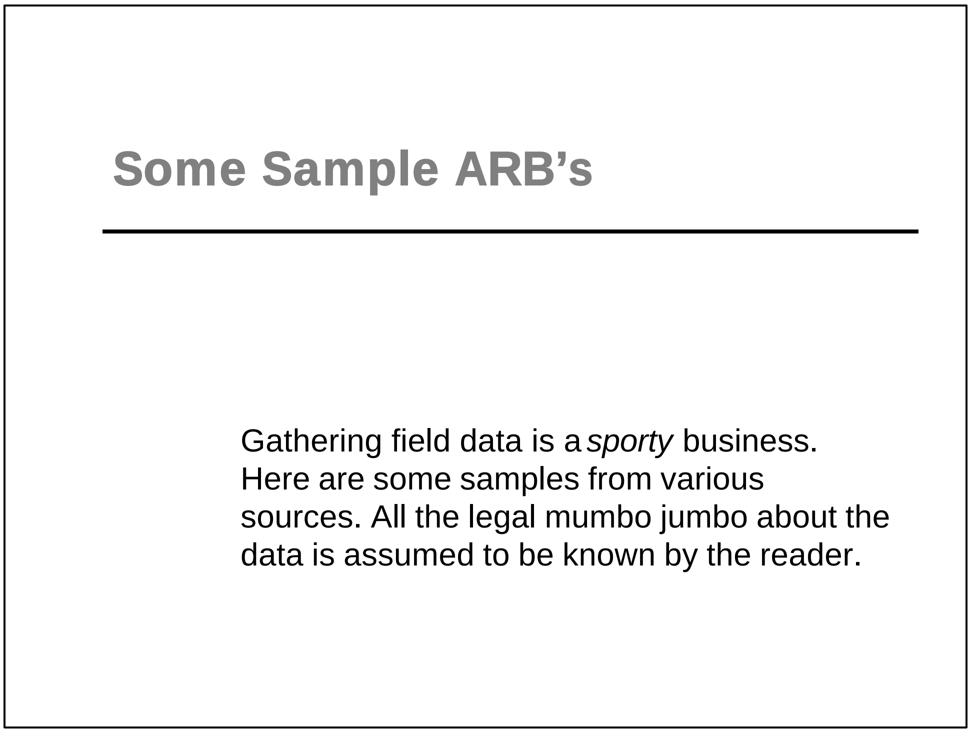
- If there are going to be quantifiable results from the ARB, some form of measurement must be put in place.
- Here are some suggestions.



ARB Summary [Abow98]

- Have a formal review with external reviewers as a planned part of the project’s lifecycle
- Time the review to best advantage. Consider an early architectural discovery review
- Choose an appropriate review technique
- Create a review contract
- Limit the number of qualities to be reviewed
- Make sure the review team includes an architectural expert, a domain expert, and support staff
- Insist on a system architect
- Collect scenarios and grow them into a check list

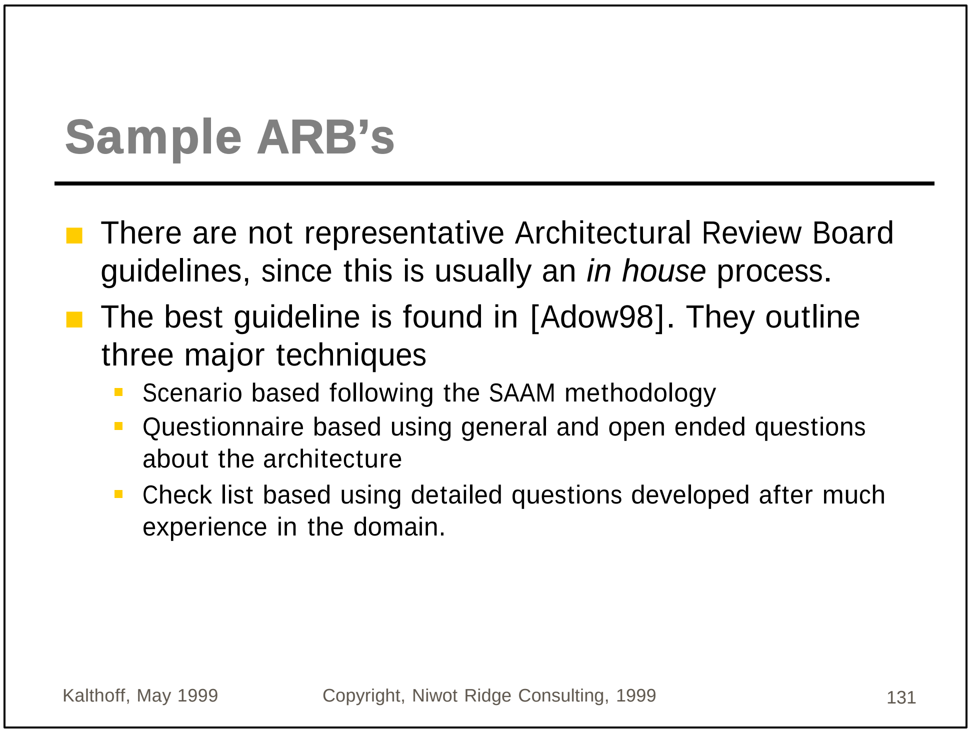
- And some more suggestions.



Some Sample ARB's

Gathering field data is a *sporty* business. Here are some samples from various sources. All the legal mumbo jumbo about the data is assumed to be known by the reader.

- One of the problems in this area of research, is there are very few case studies (that will be shared at least).
- I am fortunate to live in an area of the country where state of the art software research takes place.
- More importantly the people that perform this research are willing to share some of the results. And even more importantly there are not 10 million other people living in the same spot.

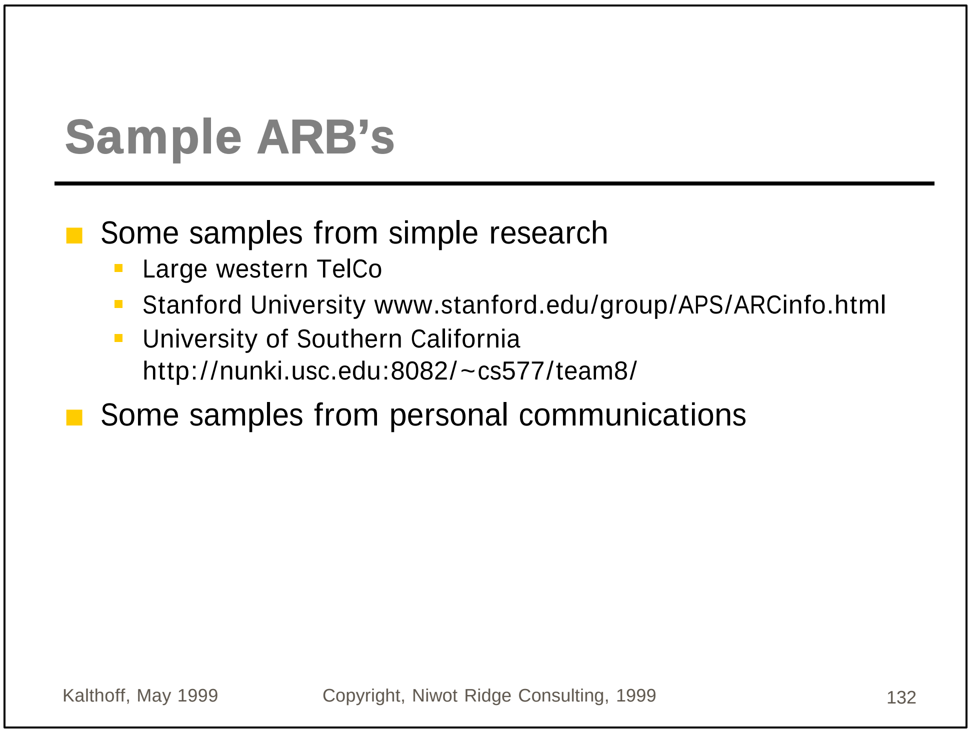


Sample ARB's

- There are not representative Architectural Review Board guidelines, since this is usually an *in house* process.
- The best guideline is found in [Adow98]. They outline three major techniques
  - Scenario based following the SAAM methodology
  - Questionnaire based using general and open ended questions about the architecture
  - Check list based using detailed questions developed after much experience in the domain.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 131

- There are some guidelines for setting up and operating an ARB, see the reference.
- But usually the ARB process is home grown, ad hoc, and dysfunctional.

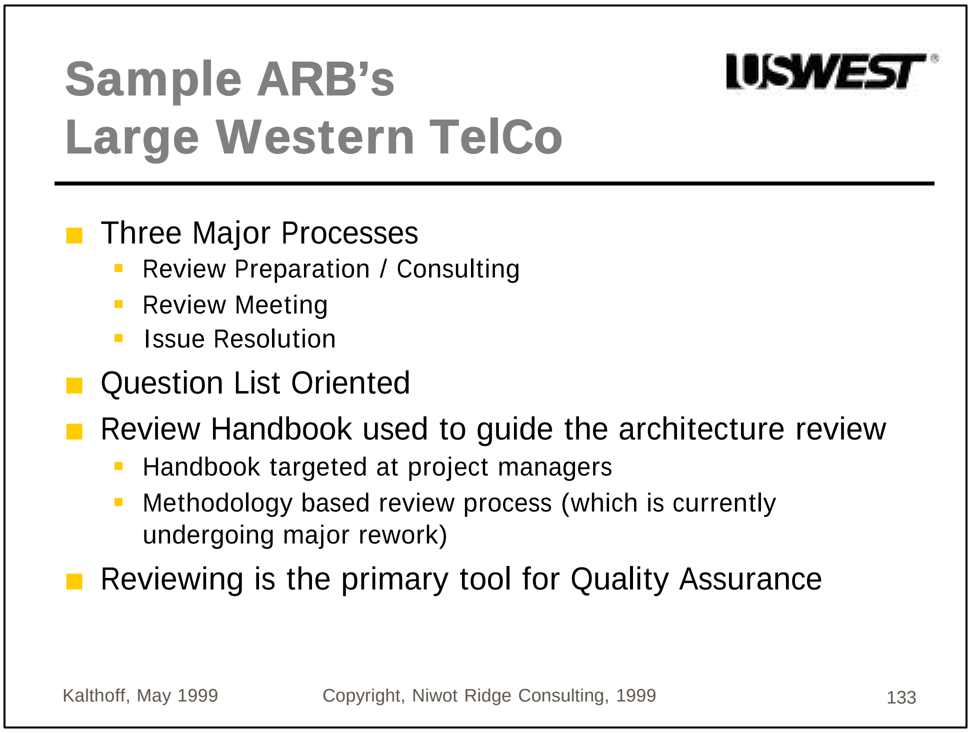


Sample ARB's

- Some samples from simple research
  - Large western TelCo
  - Stanford University www.stanford.edu/group/APS/ARCinfo.html
  - University of Southern California http://nunki.usc.edu:8082/~cs577/team8/
- Some samples from personal communications

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 132

- The actual architectural review board processes are usually *private* affairs, with little public information.
- The Web produces some information, but they are usually in public institutions.
- I have personal connections inside a regional TelCo's research center and their process was based on traditional mainframe approaches.
- With the introduction of the Web all bets are off, and they're redoing the review process as well as all the code.



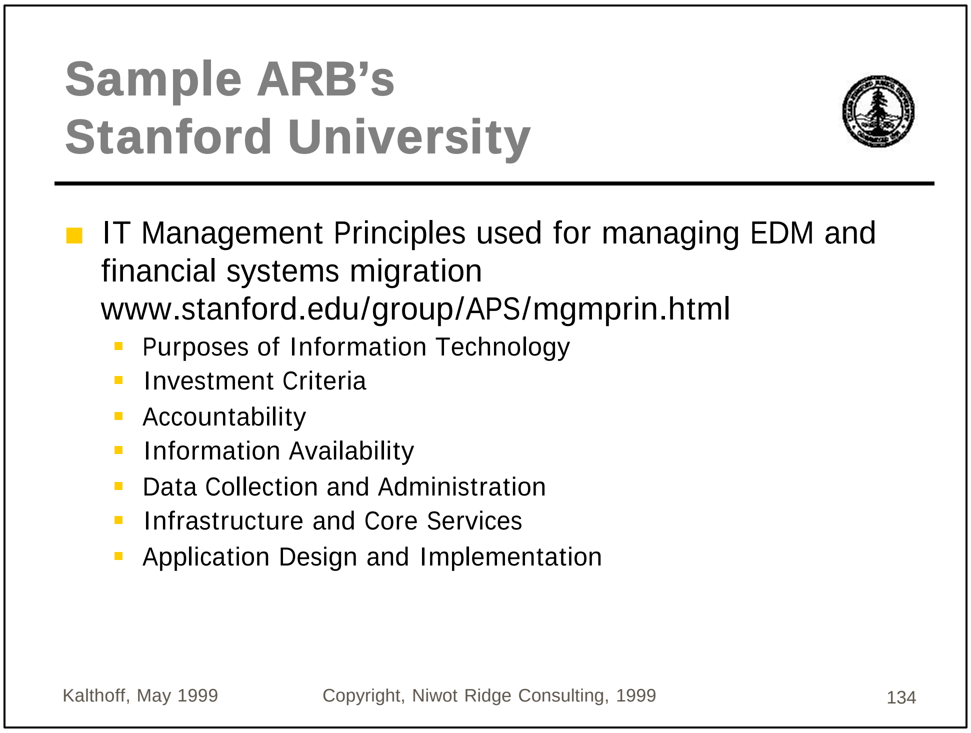
Sample ARB's Large Western TelCo

USWEST®

- Three Major Processes
  - Review Preparation / Consulting
  - Review Meeting
  - Issue Resolution
- Question List Oriented
- Review Handbook used to guide the architecture review
  - Handbook targeted at project managers
  - Methodology based review process (which is currently undergoing major rework)
- Reviewing is the primary tool for Quality Assurance

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 133

- Here is a high level view of one (and there are many) ARB processes for US West.
- It is a list oriented approach, with the experience of the participants needed to ask questions and judge the answers.

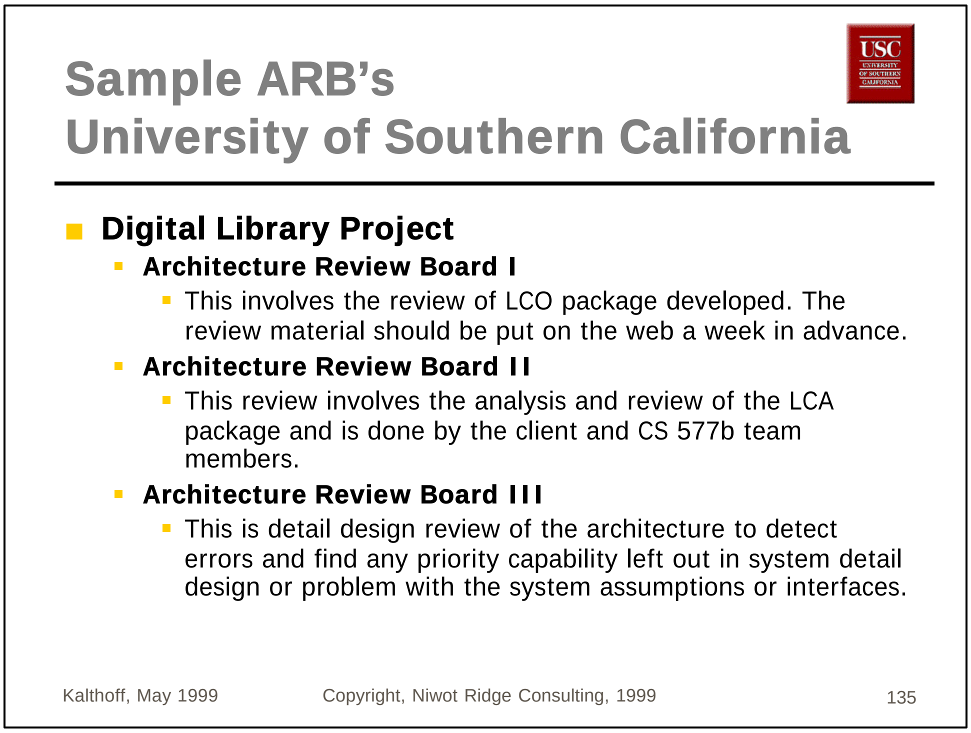


Sample ARB's Stanford University

- IT Management Principles used for managing EDM and financial systems migration www.stanford.edu/group/APS/mgmprin.html
  - Purposes of Information Technology
  - Investment Criteria
  - Accountability
  - Information Availability
  - Data Collection and Administration
  - Infrastructure and Core Services
  - Application Design and Implementation

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 134

- The Stanford approach is more structured.
- This is a very large project for converting the conventional library to a digital library.



Sample ARB's University of Southern California

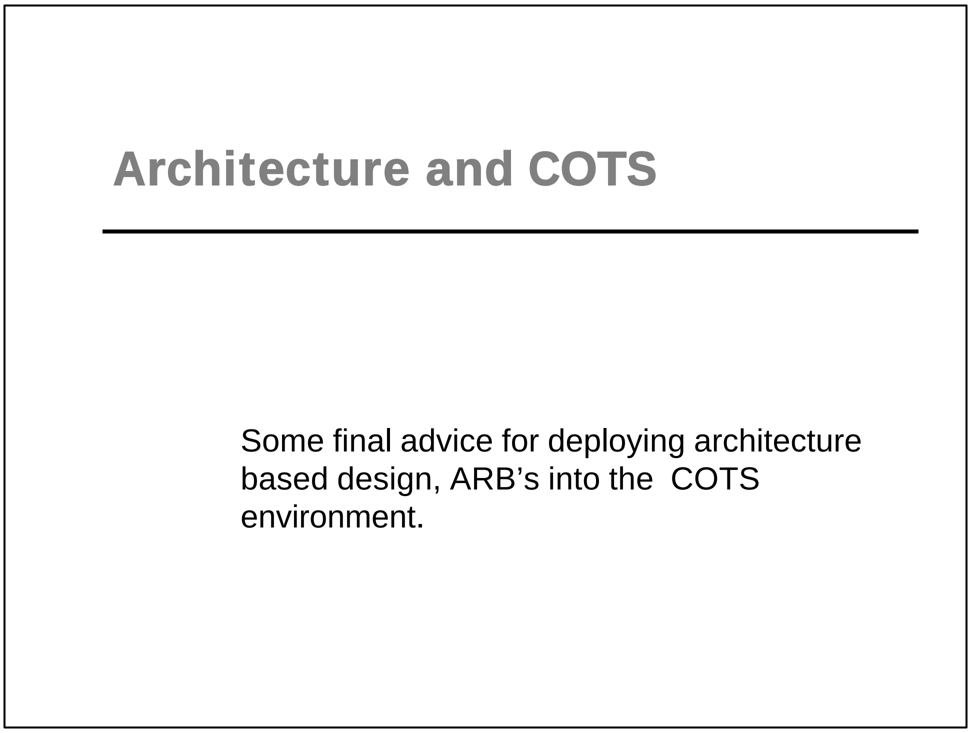
- **Digital Library Project**
  - **Architecture Review Board I**
    - This involves the review of LCO package developed. The review material should be put on the web a week in advance.
  - **Architecture Review Board II**
    - This review involves the analysis and review of the LCA package and is done by the client and CS 577b team members.
  - **Architecture Review Board III**
    - This is detail design review of the architecture to detect errors and find any priority capability left out in system detail design or problem with the system assumptions or interfaces.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 135

- Several layers of review board are in place.

Sample ARB's
University of Southern California

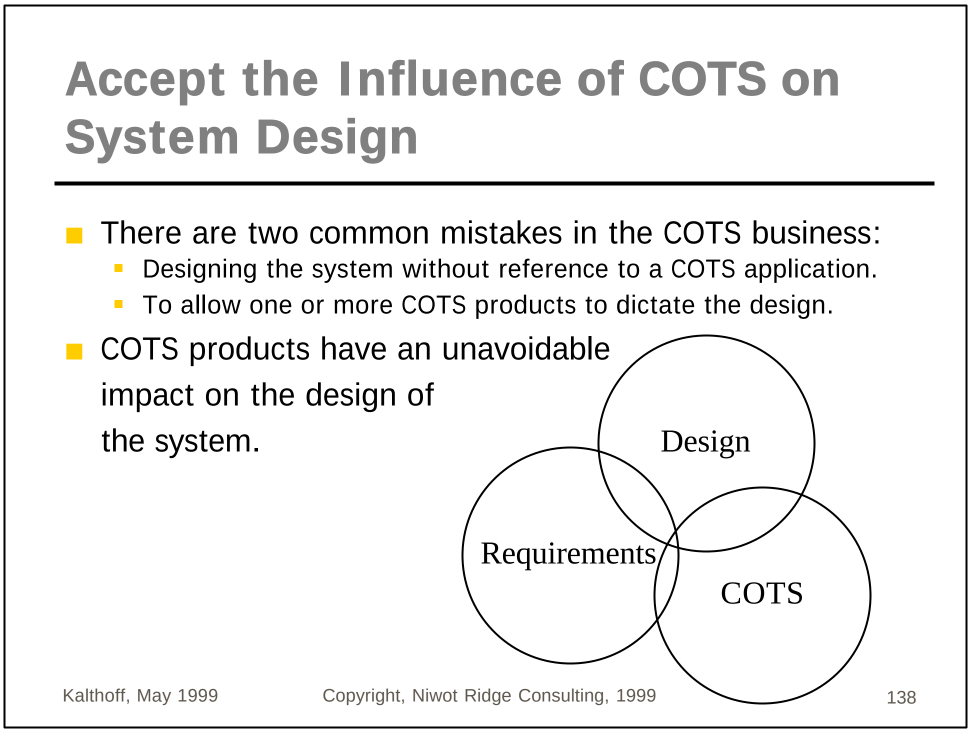
- **Digital Library Project, continued...**
  - **Reviews and Inspections**
    - In this review, different Unit Test Completion Reviews (UTCR) are performed to check that each and every module of the system has been tested and meets the specified requirements.
  - **Transition Readiness Review**
    - The Software acceptance review (SWAR) is performed here by the client along with the development team.
  - **Release Readiness Review**
    - This RRR is performed at the end of the implementation phase of the project.



Architecture and COTS

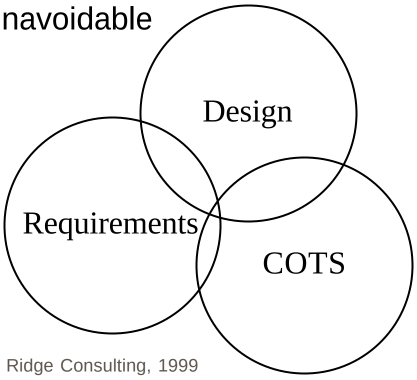
Some final advice for deploying architecture based design, ARB's into the COTS environment.

- Since we are suggesting, strongly suggesting that a COTS solution be considered, here are some impacts on the ARB.



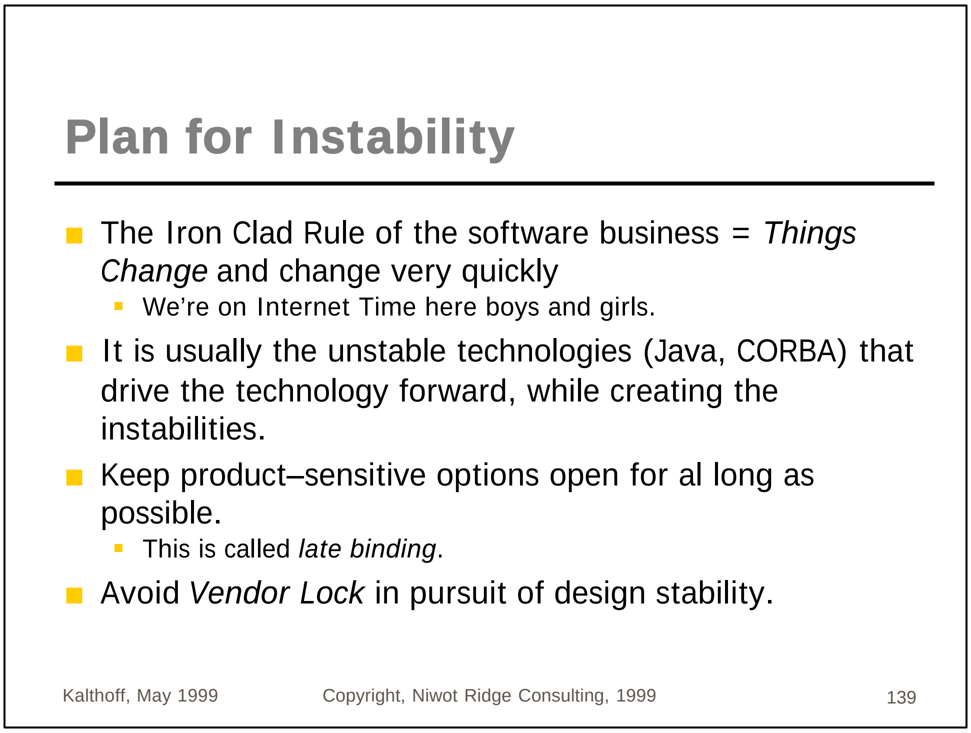
Accept the Influence of COTS on System Design

- There are two common mistakes in the COTS business:
  - Designing the system without reference to a COTS application.
  - To allow one or more COTS products to dictate the design.
- COTS products have an unavoidable impact on the design of the system.



Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 138

- When a COTS system is purchased, there is usually not much in depth analysis of the system architecture. The brochures look nice, the reference accounts all return good recommendations, etc.
- What happens though is that the user community may see the system in a different manner than the technical community.
- Even though the system is provided off the shelf, some design and architecture analysis is required before the buying decision is made.
- Remember the old transmission ad, you pay me now or you pay me later.

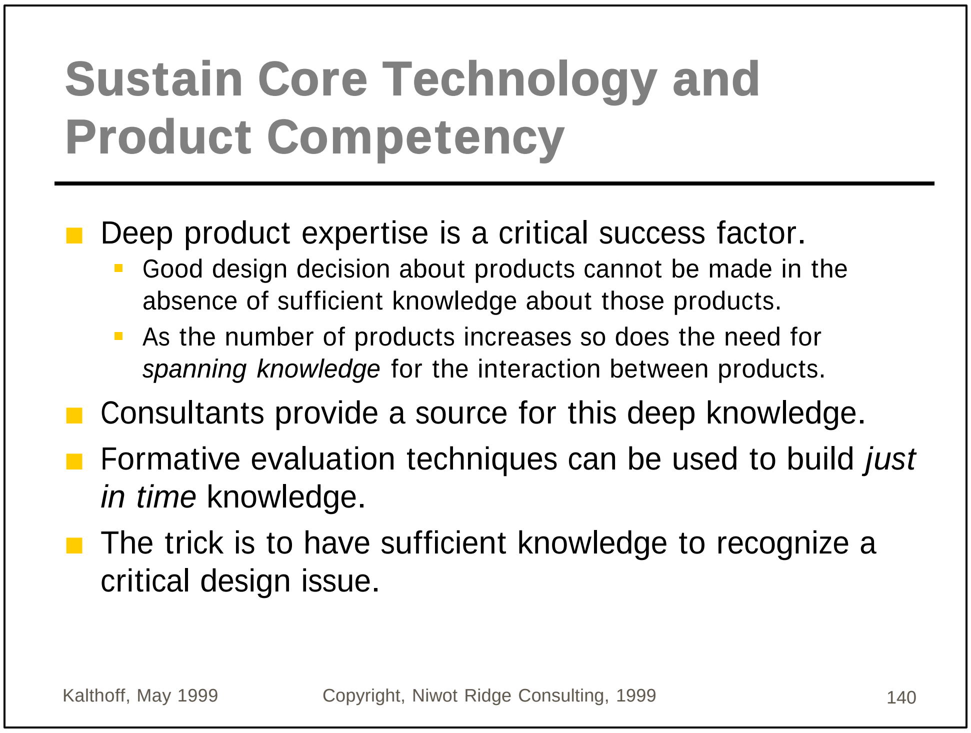


Plan for Instability

- The Iron Clad Rule of the software business = *Things Change* and change very quickly
  - We're on Internet Time here boys and girls.
- It is usually the unstable technologies (Java, CORBA) that drive the technology forward, while creating the instabilities.
- Keep product–sensitive options open for al long as possible.
  - This is called *late binding*.
- Avoid *Vendor Lock* in pursuit of design stability.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 139

- In the COTS world, change is constant.
- Add the internet time warp factor and we're in a real mess.

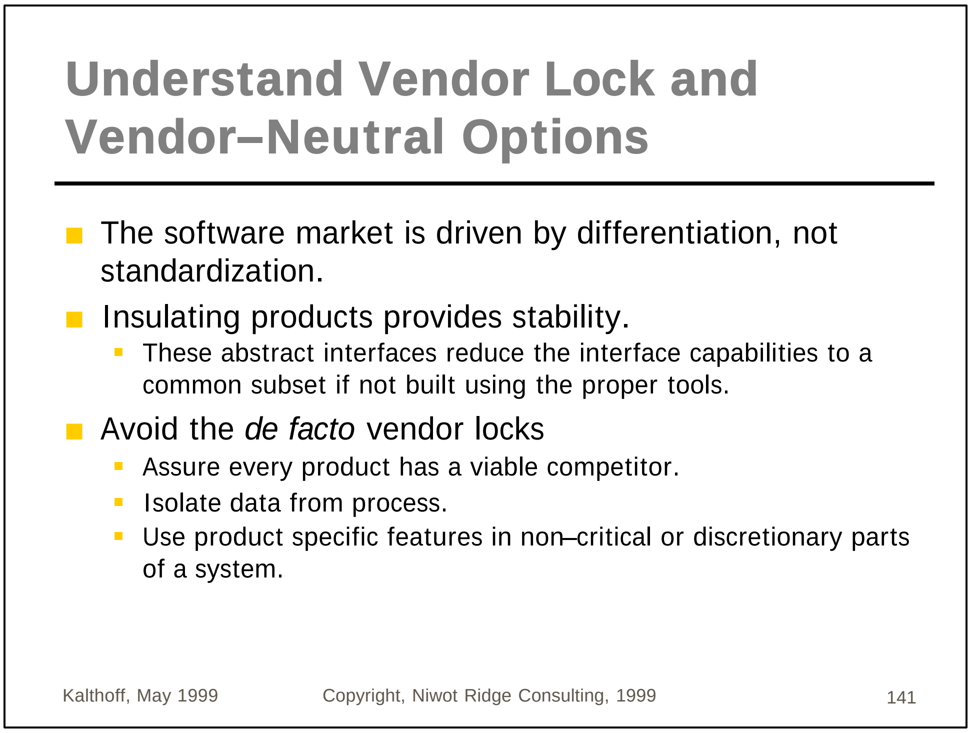


Sustain Core Technology and Product Competency

- Deep product expertise is a critical success factor.
  - Good design decision about products cannot be made in the absence of sufficient knowledge about those products.
  - As the number of products increases so does the need for *spanning knowledge* for the interaction between products.
- Consultants provide a source for this deep knowledge.
- Formative evaluation techniques can be used to build *just in time* knowledge.
- The trick is to have sufficient knowledge to recognize a critical design issue.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 140

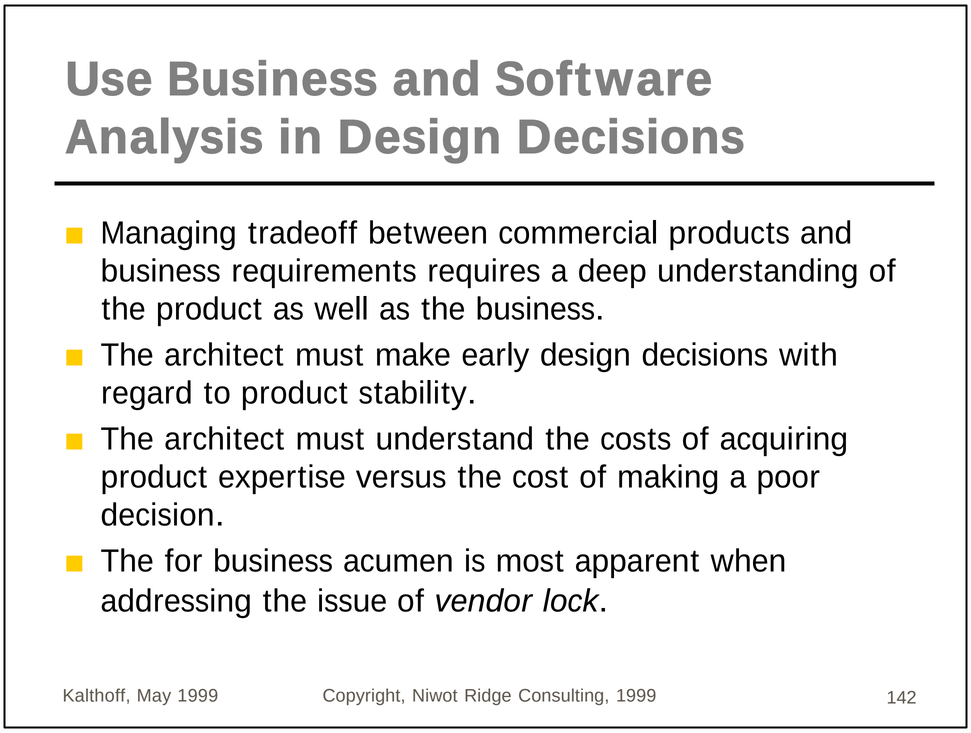
- No matter what the capabilities of the COTS application, there are core competencies that must not be displaced.



Understand Vendor Lock and Vendor–Neutral Options

- The software market is driven by differentiation, not standardization.
- Insulating products provides stability.
  - These abstract interfaces reduce the interface capabilities to a common subset if not built using the proper tools.
- Avoid the *de facto* vendor locks
  - Assure every product has a viable competitor.
  - Isolate data from process.
  - Use product specific features in non–critical or discretionary parts of a system.

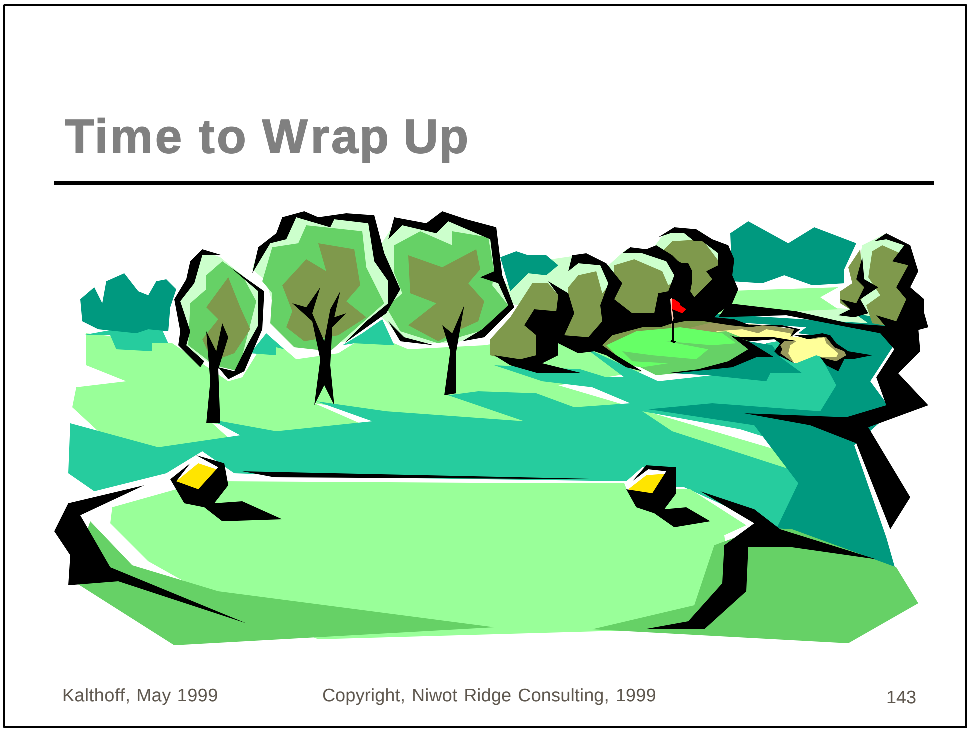
- This is a subject that is very touchy to the vendors.
- I always ask the question.
  - What if I want to get rid of your system, for what ever reason?
  - How do I move all the data and work processes to the new system?
  - The cost of this exit strategy must be factored in some way into the cost of the system.
  - The day will come when it is time to move on.
  - How much does that cost?



Use Business and Software Analysis in Design Decisions

- Managing tradeoff between commercial products and business requirements requires a deep understanding of the product as well as the business.
- The architect must make early design decisions with regard to product stability.
- The architect must understand the costs of acquiring product expertise versus the cost of making a poor decision.
- The for business acumen is most apparent when addressing the issue of *vendor lock*.

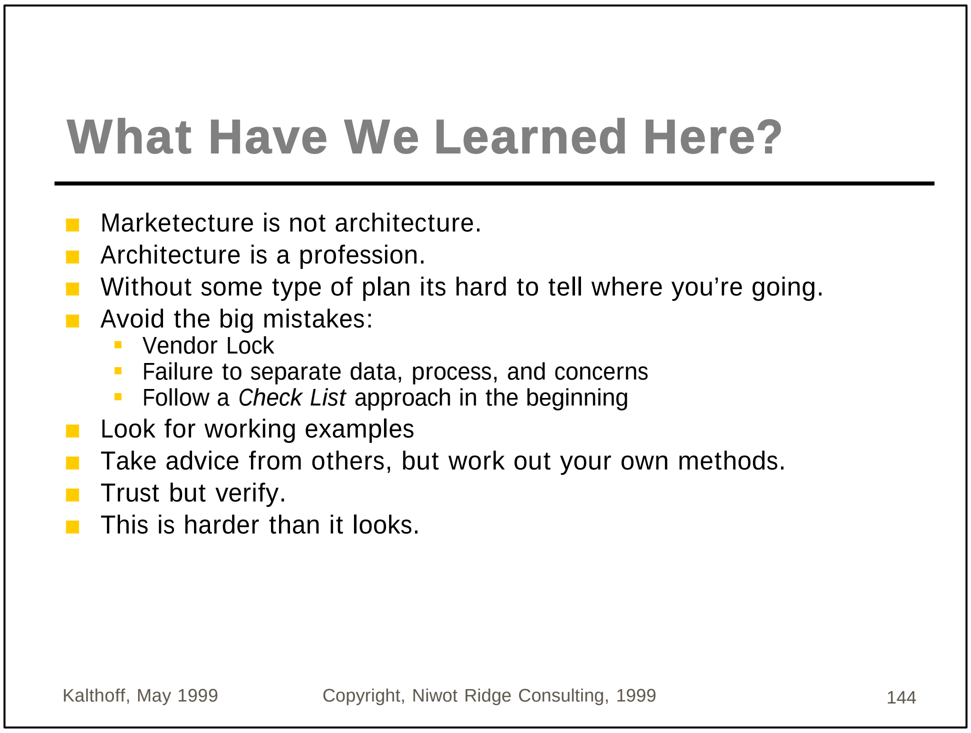
- No matter what selection process is used to COTS applications, this process must be supported by good business and design analysis.



Time to Wrap Up

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 143

- Well, its nearing the end of the session and we've all absorbed too much information. If we were in Florida, the golf bags would be standing in the lobby. Here in Philadelphia the climate is different, but the urge is the same.

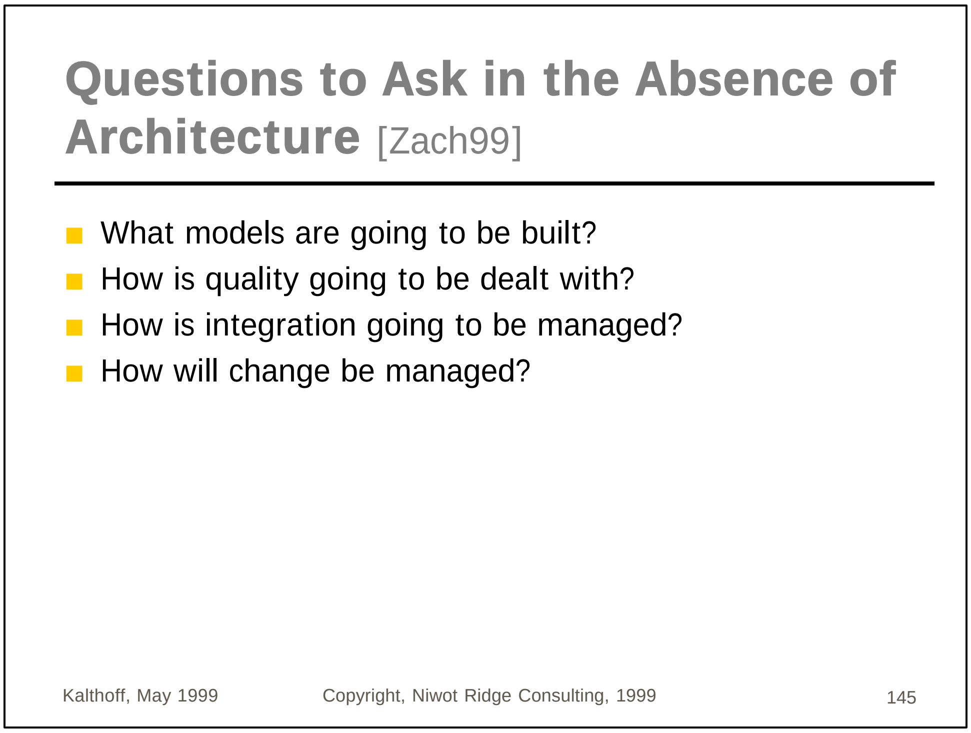


What Have We Learned Here?

- Marketecture is not architecture.
- Architecture is a profession.
- Without some type of plan its hard to tell where you're going.
- Avoid the big mistakes:
  - Vendor Lock
  - Failure to separate data, process, and concerns
  - Follow a *Check List* approach in the beginning
- Look for working examples
- Take advice from others, but work out your own methods.
- Trust but verify.
- This is harder than it looks.

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 144

- So what have we learned in the past hour or so?
- Marketecture or not architecture.
- Trust but verify.
- Avoid the big mistakes and the little ones can be addressed with your remaining money.
- Find working examples of good system architecture and copy them.
- This is actually hard work.
- The research process never stops.

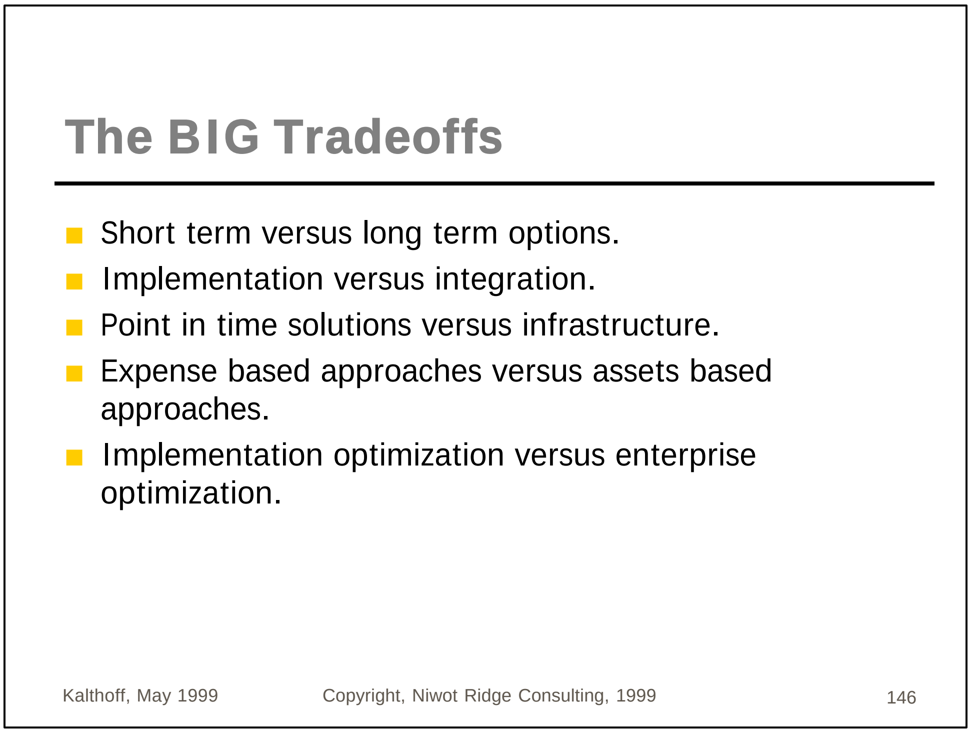


Questions to Ask in the Absence of Architecture [Zach99]

- What models are going to be built?
- How is quality going to be dealt with?
- How is integration going to be managed?
- How will change be managed?

Kalthoff, May 1999 Copyright, Niwot Ridge Consulting, 1999 145

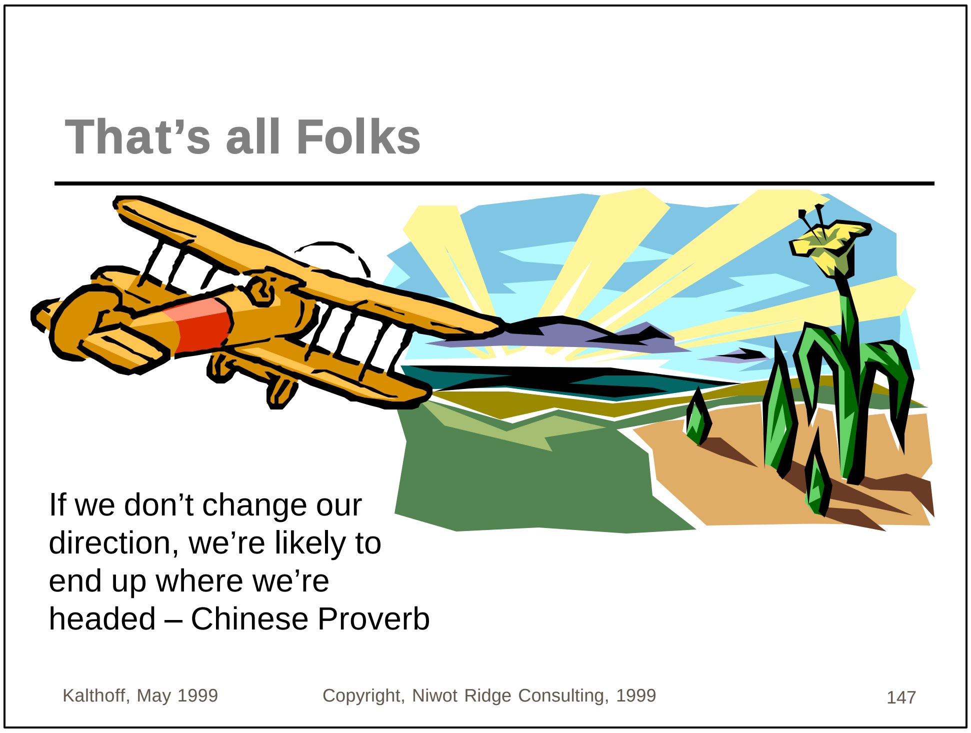
- If you're going to proceed without an architecture, here are some questions that need to be answered.



The BIG Tradeoffs

- Short term versus long term options.
- Implementation versus integration.
- Point in time solutions versus infrastructure.
- Expense based approaches versus assets based approaches.
- Implementation optimization versus enterprise optimization.

- In the end it's all about tradeoffs.
- The consequences of each of these tradeoffs can be discovered through the architectural review process.



That's all Folks

If we don't change our direction, we're likely to end up where we're headed – Chinese Proverb

Resources

In order to have any credibility, a person must have a library of knowledge upon which to draw when confronted with questions that can't be answered or when there is doubt in the mind of the audience.

Resources

- Much of the work on system architecture is available on the Web:
  - Software Engineering Institute – www.cmu.sei.org
  - Software Program Managers Network – www.spmn.com
  - Association for Computing Machinery – www.acm.org
  - Institute of Electrical and Electronics Engineering (Computer Society) – www.computer.org
  - Networked Computer Science Technical Reference Library – www.ncstrl.org
  - Worldwide Institute of Software Architects – www.wwisa.org
  - Cetus links to architecture – www.cetus-links.org/top_architecture_design.html
  - Software architecture technology guide – www-ast.tds-gn.lmco.com/arch/guide.html.

Resources

- In order for the ARB to make informed decisions, a minimum set of knowledge is necessary The following books should be read by all members of the ARB:
  - *The Art of System Architecting*, E. Rechtin and M. W. Maier, CRC Press, 1996
  - *Building Enterprise Information Architectures*, M. A. Cook, Prentice Hall, 1996
  - *Managing the Software Process*, W S Humphrey, Addison Wesley, 1989
  - *Managing Risk*, E. M. Hall, Addison Wesley, 1998
  - *Requirements Engineering: A Good Practice Guide*, I. Sommerville, John Wiley, 1997

Resources

- At some point the technical issues must be addressed These include the evaluation of the proposed system’s *abilities* The following books should be read and understood:
  - *Making Hard Decisions*, R. T. Clemen, Duxbury, 1996.
  - *Managing Multi–Objective Decisions*, M Mollaghasemi and J. Pet–Edwards, IEEE Computer Society, 1996.
  - *The Art of Computer System Performance Analysis*, R. Jain, John Wiley, 1991.
  - *The Practical Performance Analyst*, N. J. Gunther, McGraw Hill, 1998.

References

Knowledge is of two kinds; we know a subject ourselves, or we know where we can find information upon it

– Samuel Johnson

References

- [Adow95] – “Formalizing Style to Understand Descriptions of Software Architecture,” G. Abowd, G. Allen and D. Garland, *ACM Transactions*
- [Abow96] – “Recommended Best Industrial Practice for Software Architectural Evaluation,” G. Adowd, L. Bass, P. Clements, R. Kazman, L. Northrop, and A. Zaremski, *CMU/SEI–96–TR–025*.
- [Abow98] – “Architecture Reviews,” G. Abowd, L. Northrop, and A. M. Zaremski, *Software Architecture in Practice*, edited by L. Bass, P. Clements and R. Kazman, Addison Wesley, 1998.
- [Alex79] – *A Timeless Way of Building*, C. Alexander, Oxford University Press, 1979
- [Aust97] – “How to Manage ERP Initiatives,” R. D. Austin and R. L. Nolan, *Harvard Business School Working Paper*
- [Aike99] – “Reverse Engineering New Systems to Smooth Implementation,” P. Aiken and O. K. Ngwenyama, *IEEE Software*, March/April 1999, pp. 26–43.

References, continued…

- [Bass98] – *Software Architecture in Practice*, L. Bass, P. Clements and R Kazman, Addison Wesley, 1998
- [Barr95] – "IS 2000: A new Role for Corporate IS Departments and People," A. Barr, address given to *Software Development* conference, San Francisco, 1995. http://www-scip.stanford.edu/scip/avsgt/.
- [Broo75] – *The Mythical Man Month*, F. Brooks, Addison Wesley 1975.
- [Broo87] – *Report of The Defense Science Task Force on Military Software*, F. Brooks, Chairman, 1987.
- [Boar98] - *Constructing IT Blueprints*, B. Boar, John Wiley & Sons, 1998.
- [Camp98] – *The Java Tutorial: Object–Oriented Programming for the Internet*, M. Campione and K. Walrath, Sun Microsystems, 1998.
- [Busk96] – *SAP R/3 System: A Client / Server Technology*, R. Busk–Emden and J. Galimon, Addison Wesley, 1996.

References, continued...

- [Broo87] – “No Silver Bullets – Essence and Accident in Software Engineering,” F. Brooks, *IEEE Software*, April, 1997.
- [Boeh81] – *Software Engineering Economics*, B. W. Boehm, Prentice Hall, 1981.
- [Clem96] – *Coming Attractions in Software Architecture*, P. C. Clements, *CMU/SEI–96–TR–008*, Software Engineering Institute, Carnegie Mellon University, 1996.
- [Clem96a] – “Software Architecture: An Executive Overview,” P. Clements and L. Northrop, CMU/SEI–96–TR–003, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, February 1996.
- [Carl92] – “Software Measurement for DoD Systems: Recommendations for Initial Core Measures,” A. D. Carlton, Software Engineering Institute, *CMU/SEI–92–TR–19*.
- [Digr98] – “Business Object Component Architecture,” T. Digre, *IEEE Software*, Sept/Oct 1998, pp 60–69.

References, continued...

- [Dunc96] – *A Guide to the Project Management Body of Knowledge*, W. Duncan, Project Management Institute, 130 South State Road, Upper Darby, PA 19082, 1996.
- [Dvor97] – "Six Principles of High–Performance IT," R. E. Dvork, E. Holen, D. Mark, and W. F. Meehan III, *The McKinsey Quarterly*, Number 3, 1997.[Foot97] – "Big Ball of Mud," B. Foote and J. Yoder, University of Illinois at Urbana–Champaign, September 1997.
- [Garl95] – "Architectural Mismatch or Why It's Hard to Build Systems Out of Existing Parts," D. Garlan, R. Allen, and J. Ockerbloom, *Proceedings of the Seventh International Conference on Software Engineering*, April 1995.
- [Garl93] – "An Introduction to Software Architecture," D. Garlan and M. Shaw, *Advances in Software Engineering and Knowledge Engineering*, Volume 1, World Scientific, 1993.
- [Kruc95] – "The 4+1 View Model of Architecture," P. Kruchten, *IEEE Software*, 12(6), 1995.

References, continued...

- [Kazm96] – “Classifying Architectural Elements,” R. Kazman, P. Clements, G. Abowd, and L. Bass, *ACM SIGSOFT Symposium on the Foundations of Software Engineering*, 1996.
- [Laro98] – *Software Engineering Risk Management: Finding Your Path Through the Jungle*, Version 1 0, D. W. Karolak, *IEEE Computer Society*, 1998.
- [Perr92] – “Foundations for the Study of Software Architecture,” D. E. Perry and A. L. Wolf, *ACM Software Engineering Notes*, October, 1992, pp. 40–52.

References, continued…

- [Kemp98] – “Manufacturing’s Use and Abuse of IT,” R. -D. Kempis and J Ringbeck, *The McKinsey Quarterly*, Number 1, 1998.
- [DoD94] – *DoD technical Architecture Framework for Information Management (TAFIM)*, Defense Information Systems Agency (DISA) Center for Information (CIM), Volume I (Concept), Volume II (Guidance), V 2 0 Reston VA, October 1992.
- [OMG91] – *Common Object Request Broker: Architecture and Specification*, Object Management Group, Framingham, MA.
- [Plac99] - “Enterprise Solutions Structure,” E. C. Plachy and P. A. Hausler, *IBM Systems Journal*, 38(1), 1999, pp. 4-11.

References, continued…

- [SPNM99] – *Software Program Managers Network*, www.spmn.com
- [d'Sou99] – *Objects, Components, and Frameworks with UML: The Catalysis Approach*, D. F. D'Souza and A C Wills, Addison Wesley, 1999
- [ORMS97] – "RM–ODP Part 1 (Overview), Part 2 (Foundations), Part 3 (Architecture), http://www iso ch:8000/RM-ODP
- [Sowa92] – "Extending and Formalizing the Framework for Information System Architecture," Soaw and J. Zachman, *IBM Systems Journal*, 31(3), 1992
- [Tock97] – "Introduction to Zachman Framework," S. Tockey, *ORMSC/97–06–08*, Rockwell International, 1997

References, continued...

- [Adow93] – “Using Style to Understand Descriptions of Software Architecture,” G. Adowd, R. Allen and D. Garlan, *ACM Software Engineering Notes*, December, 1993.
- [Shel97] – “Adapting Zachman Framework to Business Objects,” R. Shelton, *ORMSC/97–06–09*, Open Engineering, 1997.
- [Kazm98] – “The Architecture Tradeoff Analysis Method,” R. Kazman, et al, *CMU/SEI–98–TR–008*, Software Engineering Institute, 1998.
- [Kazm99] – “Representing Software Architecture,” R Kazman, *SEI Interactive*, 12/98, http://www.sei.cmu.edu/interactive
- [McCon97] – “Software’s Ten Essentials,” S. McConnell, *IEEE Software*, March/April 1997, pp. 143–144.
- [Andr98] – “The Politics of Requirements Management,” S. Andriole, *IEEE Software*, November/December 1998, pp. 82–84.

References, continued...

- [Haum98] – “Requirements Elicitation and Validation with Real World Scenes,” P. Haumer and K. Pohl, *IEEE Transactions on Software Engineering*, 24(2), December 1998, pp. 1036–1054.
- [Gamm94] – *Design Patterns: Elements of Reusable Object–Oriented Design*, E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Addison Wesley, 1994.
- [Yell94] – “Interface, protocols, and semi–automatic construction of software adapters,” D. M. Yellin and R. E. Strom, *Proceedings of OOPLSA ‘94*, October, 1994.
- [Witt94] – *Software Architecture and Design Principles, Models, and Methods*, B I Witt, F. T. Baker, and E. W. Merritt, Van Nostrand Reinholt, 1994.

References, continued...

- [Lea93] – “Christopher Alexander: An Introduction for Object-Oriented Designers,” D. Lea, SUNY Oswego, http://g.oswego.edu/dl/ca/ca/ca.html.
- [Foot97] – “Big Ball of Mud,” B. Foote and J. Yoder, University of Illinois at Urbana–Champaign, September, 1997.
- [Shaw96] – *Software Architecture: Perspectives on an Emerging Discipline*, M Shaw, and D Garlan, Prentice–Hall, 1996.
- [Spew92] – *Enterprise Architecture Planning*, S. Spewak, John Wiley and Sons, 1992.
- [Rech97] – *The Art of Systems Architecting*, E. Rechtin and M. W. Maier, CRC Press, 1997
- [Quoi98] – “Distributed Computing Architecture,” T. Burghart, Quoion, http://www.quoininc.com.
- [Srin98] – “The Changing Role of Information Technology in Manufacturing,” K. Srinivasan and S. Jayaraman, *IEEE Computer*, March 1999, pp. 42–49.

References, continued...

- [Zach87] – “A Framework for Information Systems Architecture,” J. Zackman, *IBM Systems Journal*, **26**, Number 3, 1987.
- [Zach99] – “Life is a Series of Trade–Offs and Change is Accelerating,” J. A. Zachman, *Data to Knowledge Newsletter*, Jan/Feb, 1999.