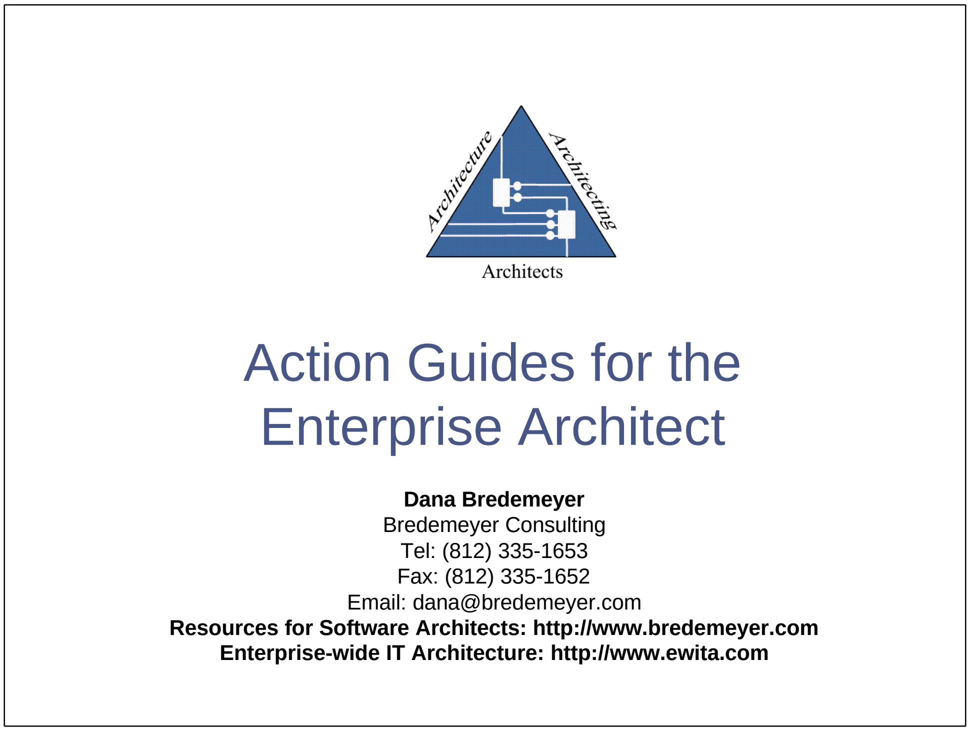

Architecture
Architecting
Architects

# Action Guides for the Enterprise Architect

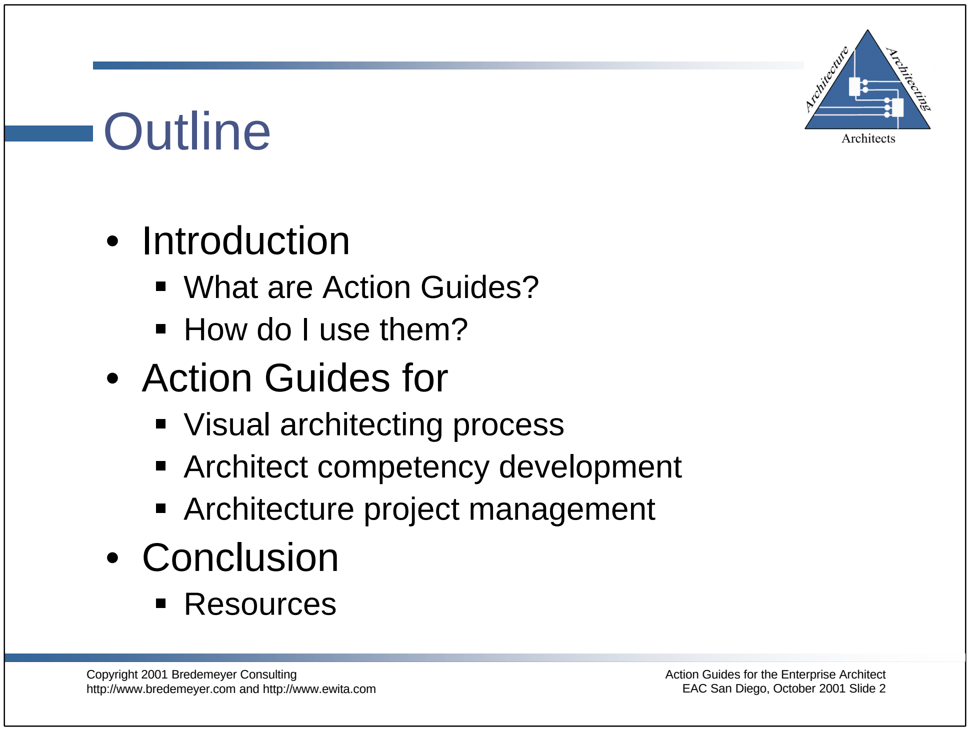
**Dana Bredemeyer**
Bredemeyer Consulting
Tel: (812) 335-1653
Fax: (812) 335-1652
Email: dana@bredemeyer.com

**Resources for Software Architects: http://www.bredemeyer.com**
**Enterprise-wide IT Architecture: http://www.ewita.com**

This is the presentation that was given by Dana Bredemeyer at the Enterprise Architecture Conference in San Diego in October, 2001.

## Acknowledgments

This talk is based on Action Guides which were created by Ruth Malan and I at Bredemeyer Consulting. Our work has been strongly influenced by work we did while at Hewlett-Packard, and by all the people we interacted with there, not to mention all the architects we have worked with since. We have integrated a number of Grove Graphics Guides (see http://www.grove.com), and use the industry-standard Unified Modeling Language (UML) for architectural modeling, with stereotyping as appropriate.



## Outline

- Introduction
  - What are Action Guides?
  - How do I use them?
- Action Guides for
  - Visual architecting process
  - Architect competency development
  - Architecture project management
- Conclusion
  - Resources

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 2

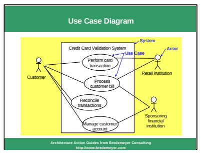
### Inspiration and Action

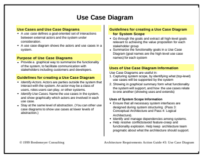
Changing architectures in organizations, whether architectures of nations or architectures of IT systems, necessitates significant organizational change. This change requires leadership. Leadership is very much about inspiring and leading action. We use stories, we use architectural vision, and we use a sense of urgency about the lack of tenability of our current state to inspire action. But vision without action does not amount to an architecture, and an architecture that is not embraced is shelf-ware.

In this talk, we focus on action, and, in particular, on the Action Guides that we have created as resources for architects faced with the challenge of moving from vision to successful evolution and deployment of architecture.

# Action Guides

- Action Guides are
  - action-oriented
  - short and fit on two sides of an 8x5 index card
    - On one side, the technique is summarized visually as a model or template
    - On the other, the following template is used:
      - – Description
      - – Purpose
      - – Notation (optional)
      - – Guidelines for Creating …
      - – Uses of … Information





### Use Case Diagram

**Use Cases and Use Case Diagrams**

- A use case defines a goal-oriented set of interactions between external actors and the system under consideration.
- A use case diagram shows the actors and use cases in a system.

**Purpose of Use Case Diagrams**

- Provide a graphical way to summarize the functionality of the system, to facilitate communication with stakeholders including customers and developers.

**Guidelines for creating a Use Case Diagram**

- *Identify Actors.* Actors are parties outside the system that interact with the system. An actor may be a class of users, roles users can play, or other systems.
- *Identify Use Cases.* Name the use cases in the system, and show graphically which actors are involved in each use case.
- Stay at the same level of abstraction. (You can other use case diagrams to show use cases at lower levels of abstraction.)

**Guidelines for creating a Use Case Diagram for System Scope**

- Go through the goals and extract all high-level goals relevant to achieving the value proposition for each stakeholder group
- Summarize the functionality goals in a Use Case Diagram (goal names are the high-level use case names) for each system

**Uses of Use Case Diagram Information**

Use Case Diagrams are useful in:

1. Capturing system scope, by identifying what (top-level) use cases will be supported by the system
2. Showing in graphical summary form what functionality the system will support, and how the use cases relate to one another (showing uses and extends)

**Uses of System Scope Information**

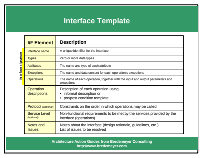
- Ensure that all necessary system interfaces are designed during system structuring (Pass 3: Conceptual Architecture and Pass 4: Logical Architecture).
- Identify and manage dependencies among systems.
- Help resolve conflicts/avoid feature-creep and functionality explosion. Help keep architecture team pragmatic about what the architecture should support.

© 1999 Bredemeyer Consulting

Architecture Requirements Action Guide #3: Use Case Diagram

# Action Guides

- Action Guides are targeted at some specific challenge that the architect faces. They may be
  - technical, e.g.
    - how to model a specific view of the architecture
    - how to document an interface
  - non-technical, e.g.
    - how to create an architecture vision
    - what to communicate to whom

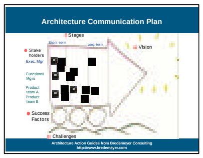


Interface Template

Interface signature

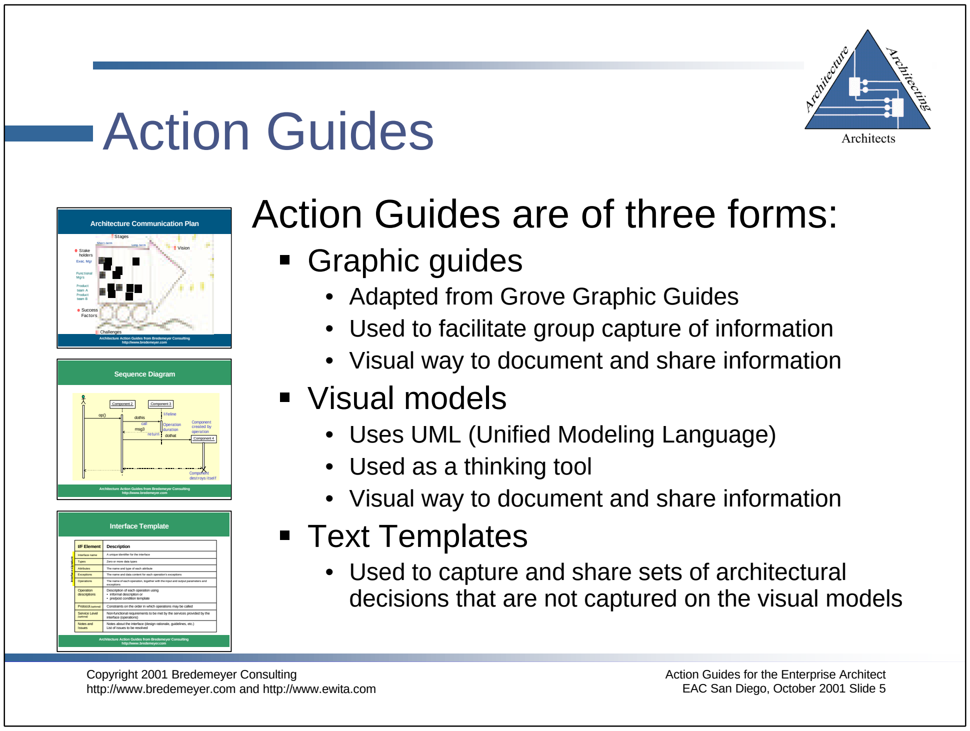
| I/F Element | Description |
|---|---|
| Interface name | A unique identifier for the interface |
| Types | Zero or more data types |
| Attributes | The name and type of each attribute |
| Exceptions | The name and data content for each operation's exceptions |
| Operations | The name of each operation, together with the input and output parameters and exceptions |
| Operation descriptions | Description of each operation using<br>• informal description or<br>• pre/post condition template |
| Protocol (optional) | Constraints on the order in which operations may be called |
| Service Level (optional) | Non-functional requirements to be met by the services provided by the interface (operations) |
| Notes and Issues | Notes about the interface (design rationale, guidelines, etc.)<br>List of issues to be resolved |

Architecture Action Guides from Bredemeyer Consulting
http://www.bredemeyer.com



Architecture Communication Plan

Architecture Action Guides from Bredemeyer Consulting
http://www.bredemeyer.com



## Action Guides

Action Guides are of three forms:

- Graphic guides
  - Adapted from Grove Graphic Guides
  - Used to facilitate group capture of information
  - Visual way to document and share information
- Visual models
  - Uses UML (Unified Modeling Language)
  - Used as a thinking tool
  - Visual way to document and share information
- Text Templates
  - Used to capture and share sets of architectural decisions that are not captured on the visual models

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 5

### Graphic Guides

Many of our graphic Action Guides are adapted from Grove Graphics Guides (see http://www.grove.com). For example, Grove's Newspaper Vision is an inspiring way to present a vision. Their Context Map is a useful way to gather and organize information about the context into which the architecture will need to fit, and we use it to identify scenarios and constraints during architectural requirements capture. We have also created a number of our own unique graphic guides, such a Stakeholder Profiles and Technology Roadmaps.  These graphic guides offer informal visual formats for gathering and presenting information. They are very useful in group situations, as working on large wall-charts to capture text and graphical icons is stimulating and facilitates arriving at group consensus.

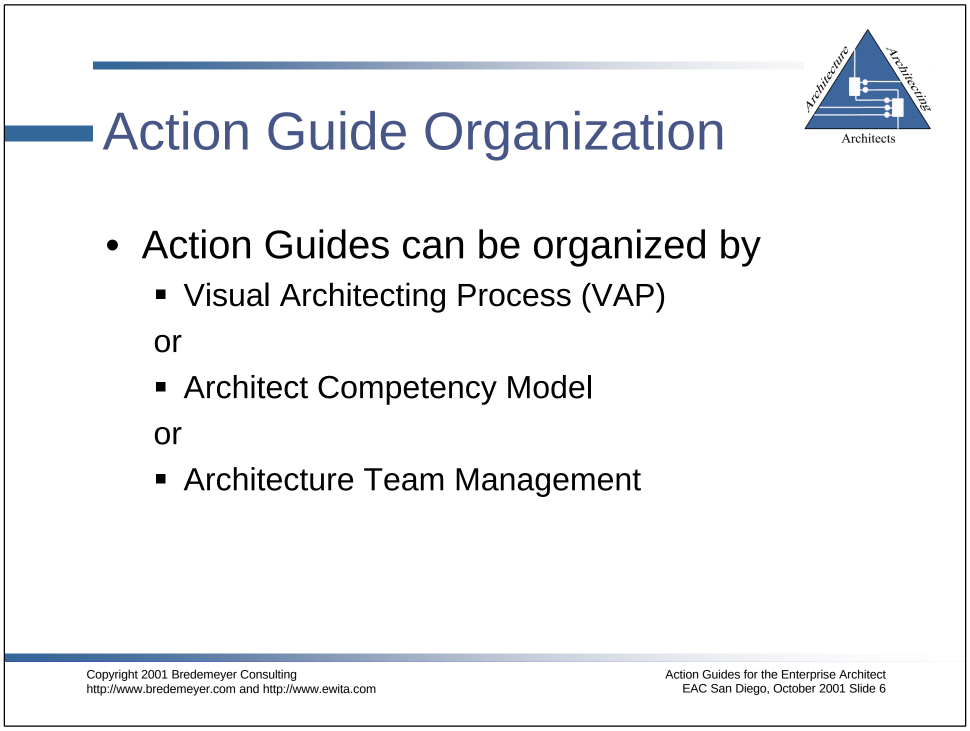
### Visual Models (UML)

Visual models are critical to communicating even complex concepts reasonably compactly. We use a subset of the industry-standard Unified Modeling Language (UML)—with stereotypes (e.g., to reflect architectural components rather than classes) where appropriate. This allows teams to use tools such as Rational Rose for architectural modeling.

### Text Templates

We provide text templates for aspects of architecture team management and for aspects of architecture decision capture.

We will see examples of each of these.



## Action Guide Organization

- Action Guides can be organized by
  - Visual Architecting Process (VAP)

  or
  - Architect Competency Model

  or
  - Architecture Team Management

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 6

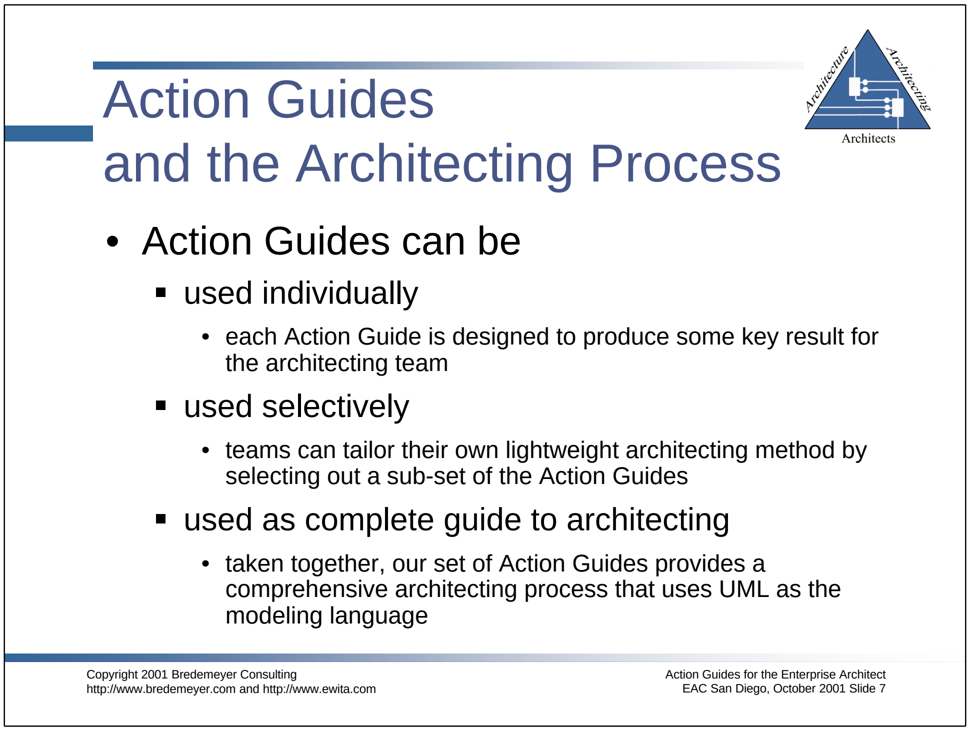
### Action Guide Organization

Given a reasonably large set of Action Guides, one needs a way to navigate through them and to pick which is appropriate to use at a given time. We offer three different organizing models:

*Visual Architecting Process (VAP):* If you are creating or updating an architecture, the Visual Architecting Process helps you identify what to do and in what general order, and has Action Guides for the different steps.

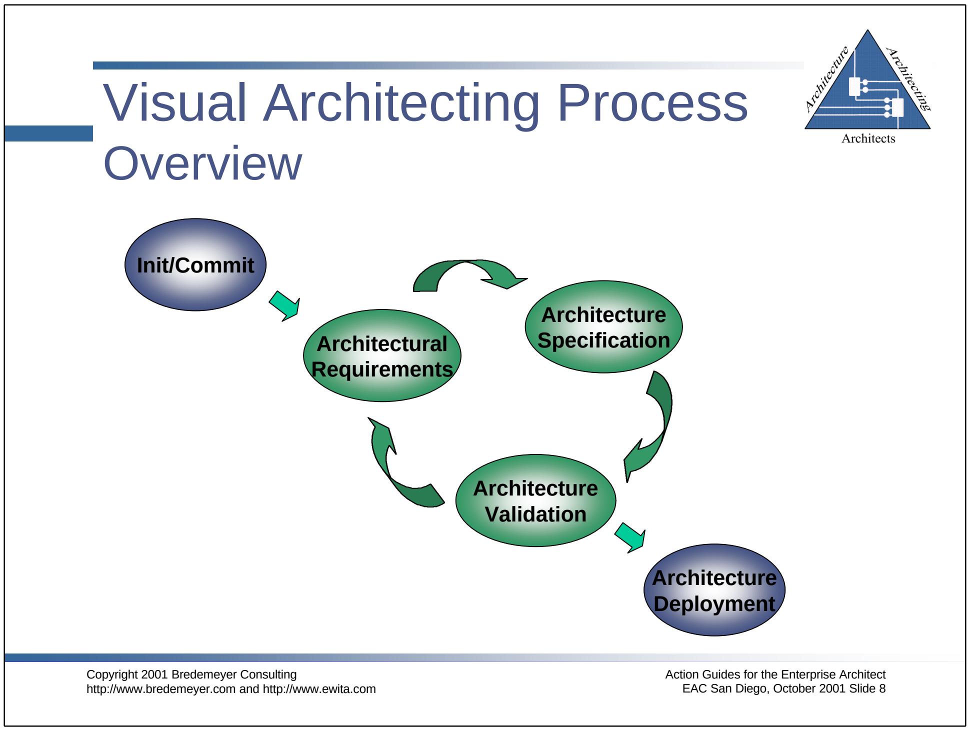
*Architect Competency Model:* This model allows you to identify your areas of strength and areas where you want to work to improve your architecting skills. You can then choose the relevant Action Guides to help you in those specific areas.

*Architecture Team Management*: We also identify Action Guides which are particularly useful to architecture managers, and when they come into play during the architecture lifecycle.



# Action Guides and the Architecting Process

- Action Guides can be
  - used individually
    - each Action Guide is designed to produce some key result for the architecting team
  - used selectively
    - teams can tailor their own lightweight architecting method by selecting out a sub-set of the Action Guides
  - used as complete guide to architecting
    - taken together, our set of Action Guides provides a comprehensive architecting process that uses UML as the modeling language



## Visual Architecting Process Overview

Init/Commit

Architectural Requirements

Architecture Specification

Architecture Validation

Architecture Deployment

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 8

## Background

The architecting process is

- based on our study of dozens of architecting projects
  - identified pitfalls and critical success factors
  - identified processes/steps that worked well
- used in workshops (since 1996)
- used (adapted) in numerous architecting projects
- continuously evolving

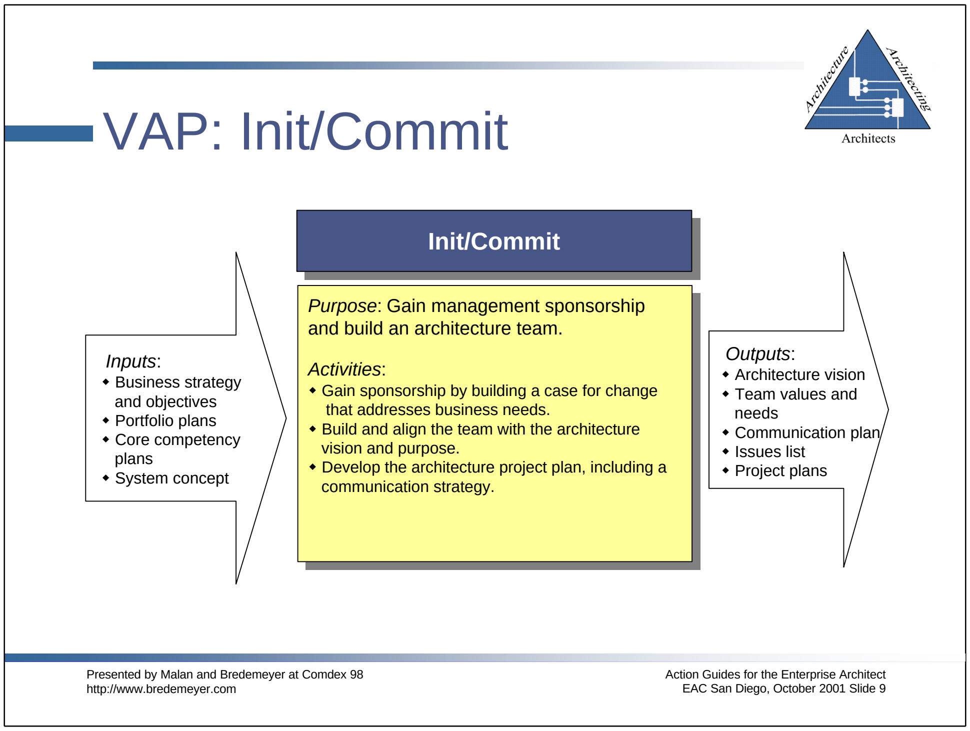
## Iteration

The architecting process is highly iterative.

First, there is picking a well-scoped problem to tackle, completing that, and then addressing another piece of the problem-space.

Second, there is iterating through cycles of Architecture Requirements, Architecture Specification and Architecture Validation, to create the layers of abstraction of the architecture (in Applications Architecture, we iterate through Meta-Architecture, Conceptual Architecture, Logical Architecture and Execution Architecture).

Third, there is picking areas that will help develop understanding of the system or which are of high risk or technical uncertainty, and driving those to detail quickly, to discover threats and opportunities and resolve critical architectural directions before exhaustively covering the full scope of the architecture specification.



## VAP: Init/Commit

**Init/Commit**

*Inputs*:
- Business strategy and objectives
- Portfolio plans
- Core competency plans
- System concept

*Purpose*: Gain management sponsorship and build an architecture team.

*Activities*:
- Gain sponsorship by building a case for change that addresses business needs.
- Build and align the team with the architecture vision and purpose.
- Develop the architecture project plan, including a communication strategy.

*Outputs*:
- Architecture vision
- Team values and needs
- Communication plan
- Issues list
- Project plans

Presented by Malan and Bredemeyer at Comdex 98
http://www.bredemeyer.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 9

## Gain management sponsorship

**Purpose:** Ensure management support through the life of the architecture project, so that management will remove obstacles to success and champion the architecture

**Activities:**

• Create/communicate the architecture vision showing how the architecture contributes to long-term business success

**Checks:**

• Do you have the resources you need?

• Are architecture team members assigned full-time?

• Does management champion the architecture vision?

## Build the architecture team

**Purpose:** Ensure a cohesive and productive team that is able to move quickly toward a sound architectural solution

**Activities:**

• Use the architecture vision to build team alignment

• Assess team capabilities and needs

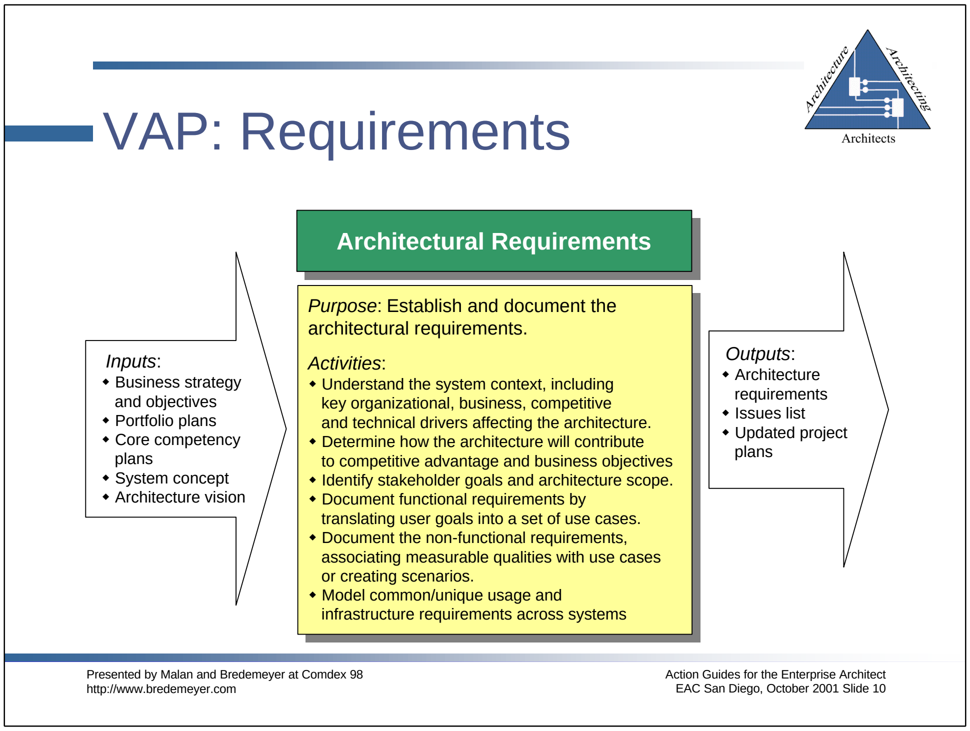
• Establish the team operating model, including team roles and responsibilities, decision model and issue resolution strategy

**Checks**:

• Is there a strong and accepted leader?

• Is the team collaborative and creative?

• Do decisions get made effectively?



## VAP: Requirements

**Architectural Requirements**

*Inputs*:
- Business strategy and objectives
- Portfolio plans
- Core competency plans
- System concept
- Architecture vision

*Purpose*: Establish and document the architectural requirements.

*Activities*:
- Understand the system context, including key organizational, business, competitive and technical drivers affecting the architecture.
- Determine how the architecture will contribute to competitive advantage and business objectives
- Identify stakeholder goals and architecture scope.
- Document functional requirements by translating user goals into a set of use cases.
- Document the non-functional requirements, associating measurable qualities with use cases or creating scenarios.
- Model common/unique usage and infrastructure requirements across systems

*Outputs*:
- Architecture requirements
- Issues list
- Updated project plans

Presented by Malan and Bredemeyer at Comdex 98
http://www.bredemeyer.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 10

## Capture Context, Goals and Scope

**Purpose:** Ensure that the architecture is aligned with the business strategy and directions, and anticipates market and technology changes

**Activities:**

• Scan the environment, identifying factors, trends and  forces that are likely to impact the architecture

• Establish which business objectives apply to the  architecture to ensure that the architecture is aligned with the business agenda

• Determine where the architecture will provide competitive differentiation, and where not

• Elicit and record stakeholder goals to discover opportunities to support stakeholder goals and focus on providing specific perceived benefit to the stakeholder

• Determine the architecture scope to provide focus and direction and identify dependencies

## Capture Functional Requirements

**Purpose:** Document, communicate and validate the intended behavior of the system

**Activities:**

• Use *use cases* to capture *who* (actor) does *what* (interaction) with the system, for what *purpose* (goal), without dealing with system internals

## Capture Non-functional Requirements

**Purpose:** Explicit, documented non-functional requirements are needed to: define architectures so that they achieve the required qualities; form a basis for comparing alternatives and make tradeoffs; enhance communication; and evaluate the architecture

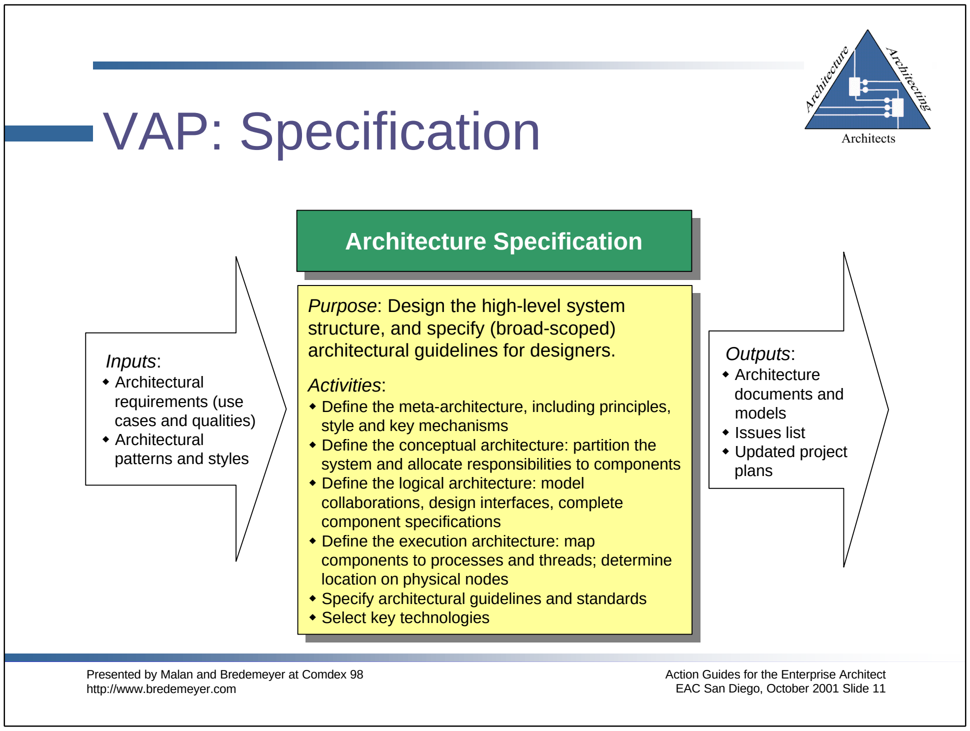
**Activities:**

• Identify non-functional requirements (qualities and constraints)

• Define each required quality attribute unambiguously

• State a measure or test that will be used to ensure that the quality attribute is met

• Prioritize the non-functional requirements



## VAP: Specification

**Architecture Specification**

*Inputs*:
- Architectural requirements (use cases and qualities)
- Architectural patterns and styles

*Purpose*: Design the high-level system structure, and specify (broad-scoped) architectural guidelines for designers.

*Activities*:
- Define the meta-architecture, including principles, style and key mechanisms
- Define the conceptual architecture: partition the system and allocate responsibilities to components
- Define the logical architecture: model collaborations, design interfaces, complete component specifications
- Define the execution architecture: map components to processes and threads; determine location on physical nodes
- Specify architectural guidelines and standards
- Select key technologies

*Outputs*:
- Architecture documents and models
- Issues list
- Updated project plans

Presented by Malan and Bredemeyer at Comdex 98
http://www.bredemeyer.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 11

### Create the Meta-Architecture

**Purpose:** Make strategic architectural choices to guide the architecting effort

**Activities:**

• Review other architectures, styles and patterns and gather lessons from past experience

• Create architectural principles. Select/adapt applicable architectural style(s) or patterns.

• Decide on concepts and mechanisms to ensure  architectural integrity and consistency

### Create the Conceptual Architecture

**Purpose:** Create conceptual models to communicate the architecture to management sponsors, project managers for team/individual work assignments and customers/users. Also allows for early validation of key architectural decisions and forms the starting point for the logical architecture.

**Activities:**

• Create an architecture diagram showing the system decomposition into components and connectors

• Create informal component specifications, documenting each component's responsibilities, the components it collaborates with in accomplishing the responsibilities and the rationale for clustering the responsibilities in that component

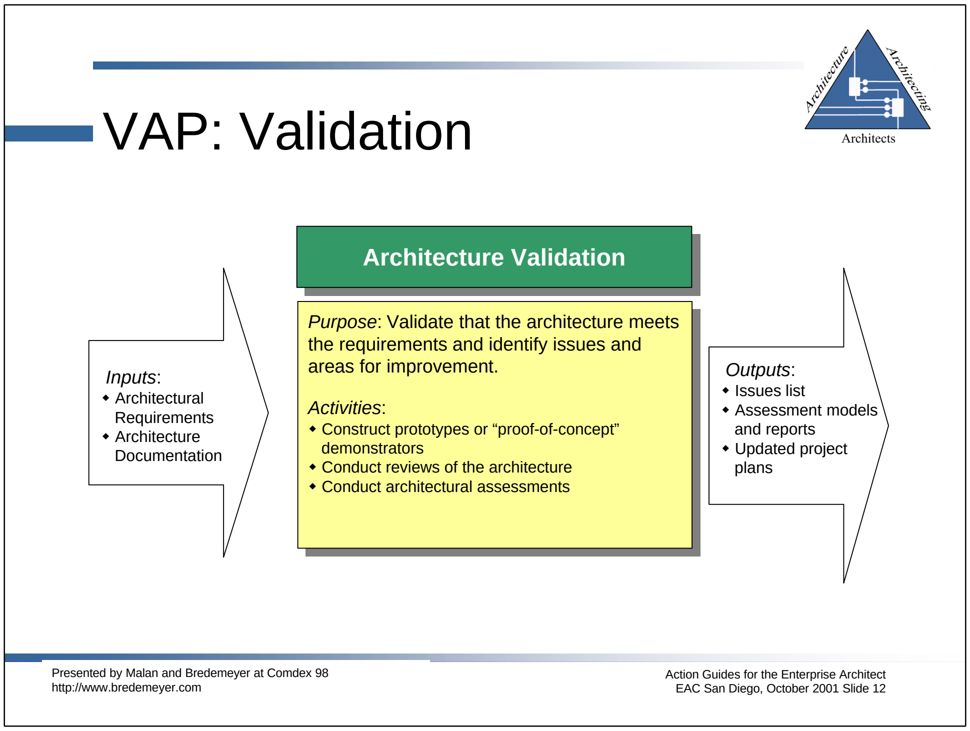
### Create the Logical Architecture

**Purpose:** Create detailed architectural specifications to document the architecture decisions and to communicate the architecture to designers, developers and contractors in a way that is directly actionable, clear and unambiguous

**Activities:**

• Use component collaboration diagrams (CCD) to explore and document system behavior--helpful in elaborating the component interfaces.

• Create detailed component specifications, documenting the interfaces (list of operations, descriptions of the operations, constraints represented as pre-post conditions on the operations or state diagrams, etc.) and component use model (concurrency model, constraints on component composition, lifecycle model, how the component is instantiated, how it is named, a test or performance suite, etc.)



# VAP: Validation

**Architecture Validation**

*Inputs*:
- Architectural Requirements
- Architecture Documentation

*Purpose*: Validate that the architecture meets the requirements and identify issues and areas for improvement.

*Activities*:
- Construct prototypes or “proof-of-concept” demonstrators
- Conduct reviews of the architecture
- Conduct architectural assessments

*Outputs*:
- Issues list
- Assessment models and reports
- Updated project plans

Presented by Malan and Bredemeyer at Comdex 98
http://www.bredemeyer.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 12

## Validate the Architecture

**Purpose:** Assess the architecture to validate that it meets the requirements and identify issues and areas for improvement early

**Activities:**

• Construct prototypes or “proof-of-concept” demonstrators  or build a skeletal architecture to validate communication and control mechanisms and interfaces

• Conduct reviews of the architecture to check that principles are upheld and other meta-architecture guidelines are met, and discuss how the architecture meets the goals and requirements placed on it

• Conduct architectural assessments to assess the architecture against use cases to see that it will support the required functionality, and scenarios to see that the system qualities are met (see the SAAM Action Guide*)

**Checks**:

*i. Goodness of the architecture*

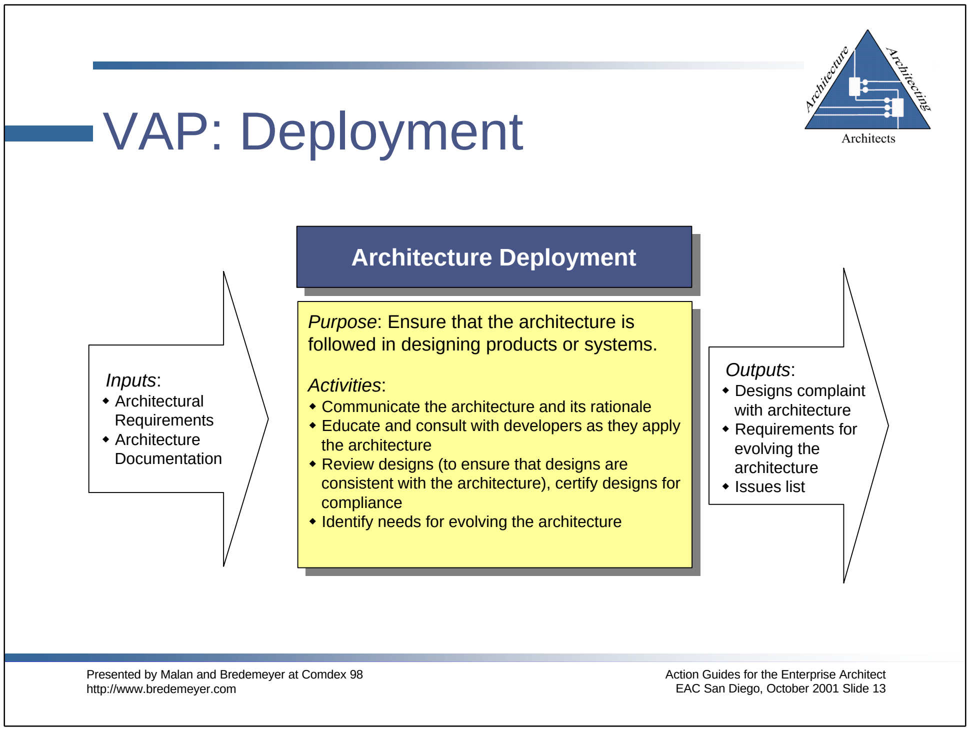
• Did you satisfy yourself and others that the architecture as defined satisfies the stakeholder goals and requirements?

• Did you assess the conceptual integrity, correctness, and buildability of the architecture?

• Did you assess the degree to which the architecture will flex to meet future requirements?

*ii. Goodness of the architecture documentation*

• Did you assess the understandability and utility of the documentation?



# VAP: Deployment

**Architecture Deployment**

*Inputs*:
- Architectural Requirements
- Architecture Documentation

*Purpose*: Ensure that the architecture is followed in designing products or systems.

*Activities*:
- Communicate the architecture and its rationale
- Educate and consult with developers as they apply the architecture
- Review designs (to ensure that designs are consistent with the architecture), certify designs for compliance
- Identify needs for evolving the architecture

*Outputs*:
- Designs complaint with architecture
- Requirements for evolving the architecture
- Issues list

Presented by Malan and Bredemeyer at Comdex 98
http://www.bredemeyer.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 13

## Build Understanding

**Purpose:** Help the developer community understand the architecture, its rationale, and how to use it. Help the management community understand its implications for organizational success, work assignments, etc.

**Activities:**

- “Communicate, communicate, communicate” (Rechtin, 1996). Listen! Give presentations, keep an “open door”,  write good documentation, etc.
- Consult: always be available to assist and consult
- Educate: create tutorials and demos

## Ensure Compliance

**Purpose:** Ensure that designs and implementations adhere to the architecture and do not cause architectural drift

**Activities:**

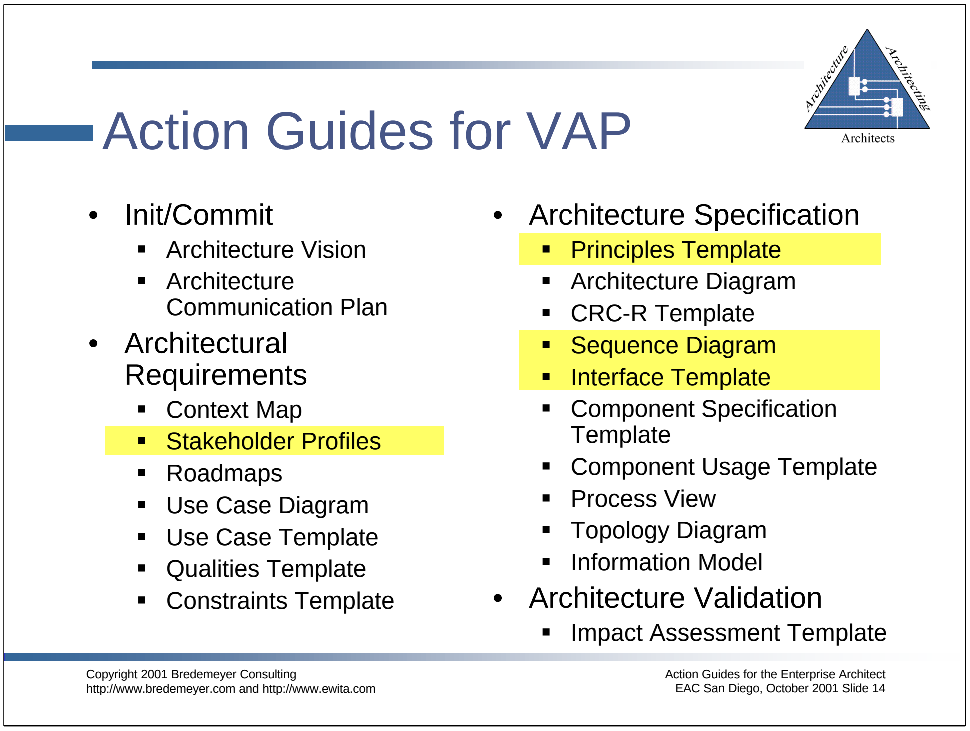
- Review designs to ensure they are consistent with the architecture

## Evolve the Architecture

**Purpose:** Ensure that the architecture remains current

**Activities:**

- Actively watch for and respond to need for changes to the  architecture. Stay engaged!



## Action Guides for VAP

- Init/Commit
  - Architecture Vision
  - Architecture Communication Plan
- Architectural Requirements
  - Context Map
  - Stakeholder Profiles
  - Roadmaps
  - Use Case Diagram
  - Use Case Template
  - Qualities Template
  - Constraints Template
- Architecture Specification
  - Principles Template
  - Architecture Diagram
  - CRC-R Template
  - Sequence Diagram
  - Interface Template
  - Component Specification Template
  - Component Usage Template
  - Process View
  - Topology Diagram
  - Information Model
- Architecture Validation
  - Impact Assessment Template

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 14

### Action Guides for Enterprise Architecture

Enterprise architecture is the high-level structure of enterprise systems. It can be viewed as a layered model:

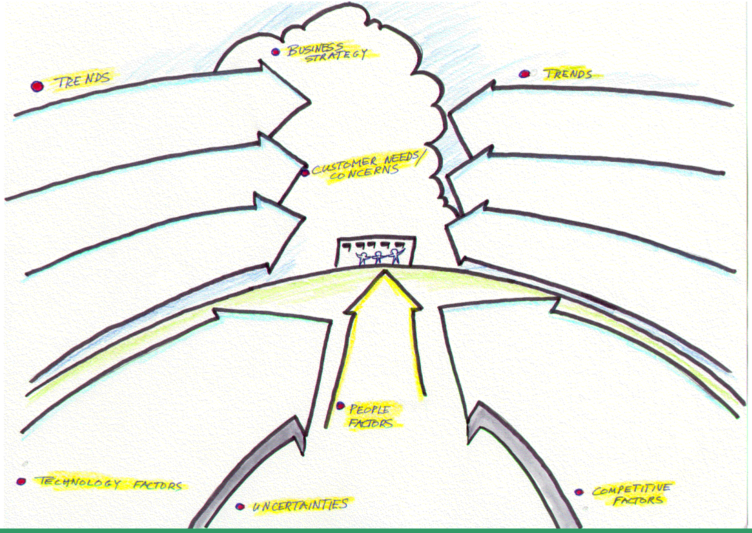
- Business processes and supporting organizational structure
- Applications that support the business processes
- Infrastructure that supports the applications
- Information that supports the applications

This layered model decomposes enterprise architecture in a way that is useful from an organizational point of view. That is, each of these layers is addressed by specialists: business architects, application/software architects, technology architects and information/data architects. However, whenever a system is decomposed, you have to pay special attention to the relationships among the components and the interfaces between them. You also need to have a senior architect who leads the overall architecting effort and oversees issues and decisions whose scope crosses the boundaries of the individual components.

That said, the **Visual Architecting Process (VAP)**, applies in a generalized way to each of the sub-disciplines of enterprise architecture. However, in creating the architecture specification for the various sub-disciplines, different models and views--and hence action guides--are relevant. Thus, Interface Templates are central to putting teeth into an application architecture specification. Information models are created by information architects and used by application architects when designing component interfaces.  Topology Diagrams are created by technology architects to document and design the hardware and software infrastructures upon which the software systems will be deployed. These are annotated by application architects to reflect where application components will execute.  Etc.

The above list of Action Guides is a representative but *not* exhaustive set. For example, there are a number of Action Guides to help in creating architecture strategy. What distinguishes, for example, the Applications layer/view of Enterprise Architecture from the architecture of an individual application, is that the scope of Enterprise Architecture crosses (internal) organizational boundaries--because of this broad scope, Enterprise Architects have to pick their targets very strategically, or spin their wheels trying to tackle too much.

# Context Map



**Architecture Action Guides from Bredemeyer Consulting**
**http://www.bredemeyer.com**

# Context Map

### Purpose of Context Map Template

- Facilitate conducting an environmental scan, providing reminders of important categories to consider
- Provide effective graphical group communication medium.

### Guidelines for creating a Context Map

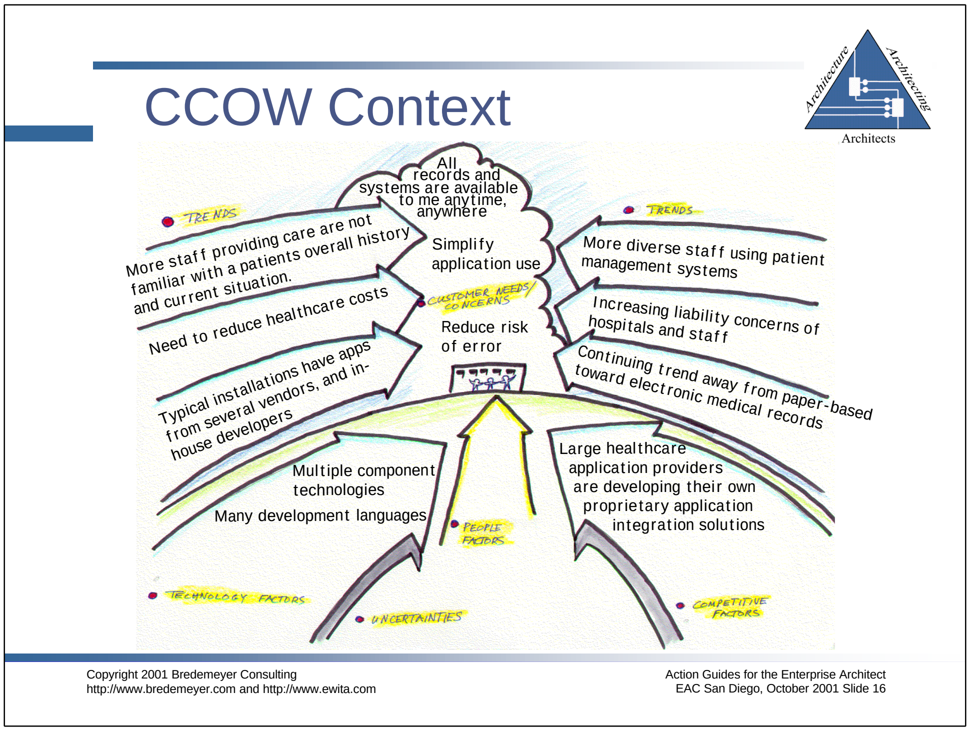
1. *Identify/agree on categories*: Identify and get agreement on the most relevant categories of environmental information. Add to or change the Guide accordingly.
2. *Identify trends*: Record technological and industry/market trends.
3. *Identify drivers/forces and factors*: Record competitor directions, partner/supplier directions, customer needs, governmental regulations, people/resource needs and constraints and other forces or factors that might affect the architecture.
4. *Identify areas of uncertainty*
5. *Identify issues and risks*

### Uses of Context Map information

- Use context map to identify and state assumptions about business and market directions and technology trends that the architecture will need to accommodate.
- Starting point for identifying change scenarios when capturing non-functional requirements (What new business processes or behaviors, applications or services does the new technology enable? What new capabilities will the architecture have to support? What competitor moves will the business need to respond to, and how will the architecture help?

❋ Adapted from Grove Consultants International's Graphic Guide #3 (See http://www.grove.com)

Init/Commit Action Guide #1: Cover-Story Vision



# CCOW Context

All records and systems are available to me anytime, anywhere

Simplify application use

Reduce risk of error

More staff providing care are not familiar with a patients overall history and current situation.

Need to reduce healthcare costs

Typical installations have apps from several vendors, and in-house developers

More diverse staff using patient management systems

Increasing liability concerns of hospitals and staff

Continuing trend away from paper-based toward electronic medical records

Multiple component technologies

Many development languages

Large healthcare application providers are developing their own proprietary application integration solutions

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

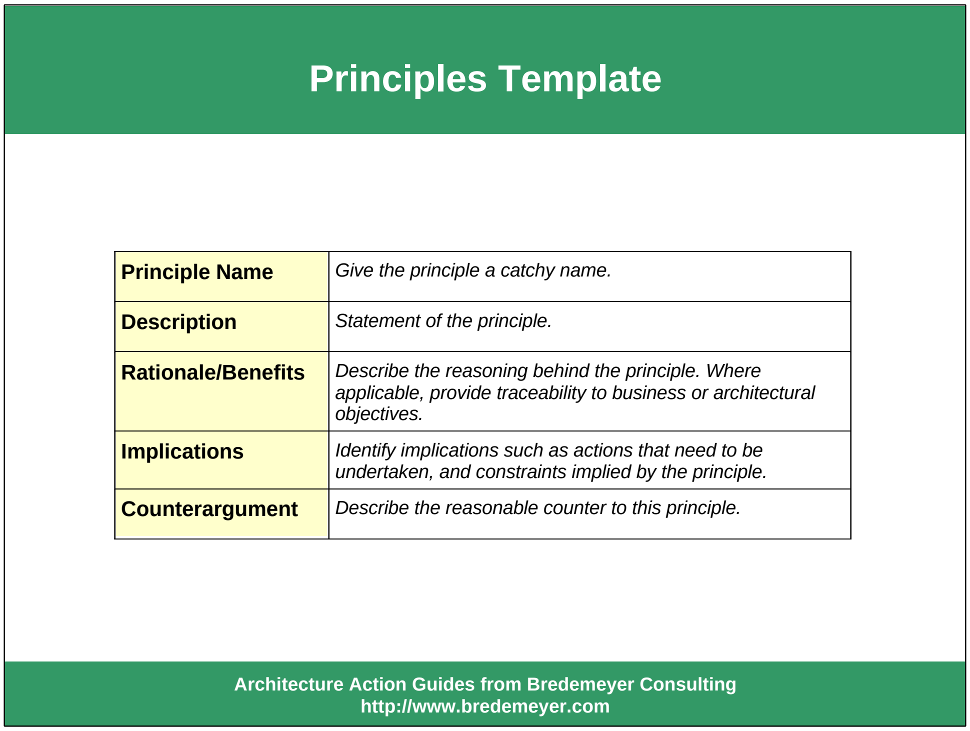
Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 16

## CCOW

Common Clinical Context (CCOW) Architecture  Specification (ver. 1.1) available from HL7 at http://www.hl7.org

### CCOW Vision

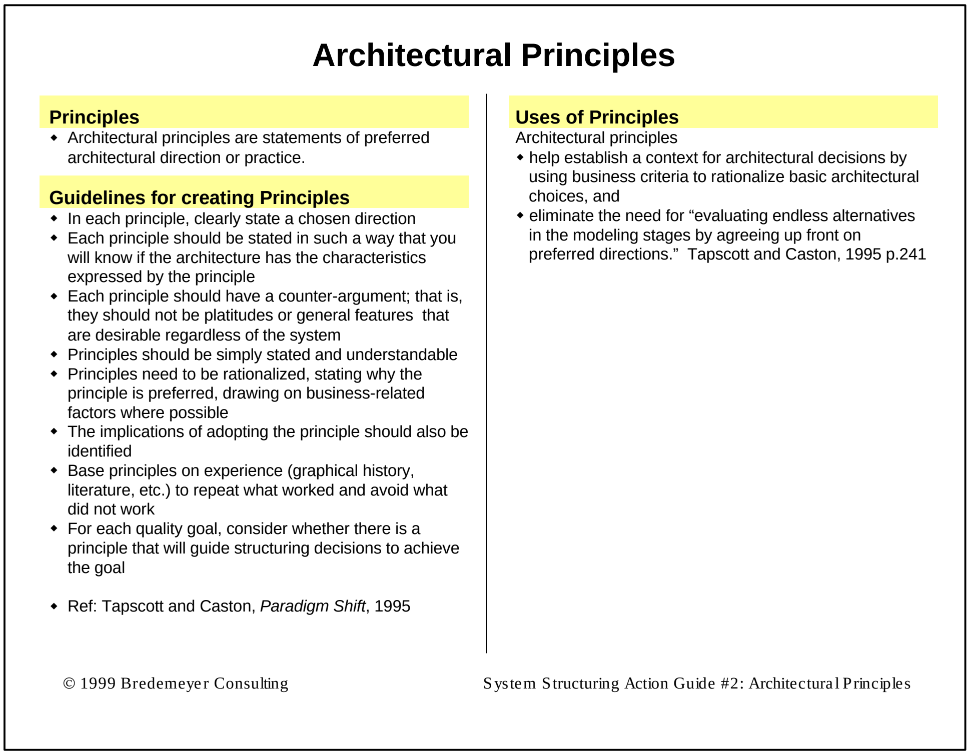
The user of a set of independently developed healthcare applications running on a single desktop is able to act as if she is interacting with a single integrated application.



## Principles Template

| | |
|---|---|
| **Principle Name** | *Give the principle a catchy name.* |
| **Description** | *Statement of the principle.* |
| **Rationale/Benefits** | *Describe the reasoning behind the principle. Where applicable, provide traceability to business or architectural objectives.* |
| **Implications** | *Identify implications such as actions that need to be undertaken, and constraints implied by the principle.* |
| **Counterargument** | *Describe the reasonable counter to this principle.* |

**Architecture Action Guides from Bredemeyer Consulting**
**http://www.bredemeyer.com**



## Architectural Principles

### Principles

- Architectural principles are statements of preferred architectural direction or practice.

### Guidelines for creating Principles

- In each principle, clearly state a chosen direction
- Each principle should be stated in such a way that you will know if the architecture has the characteristics expressed by the principle
- Each principle should have a counter-argument; that is, they should not be platitudes or general features that are desirable regardless of the system
- Principles should be simply stated and understandable
- Principles need to be rationalized, stating why the principle is preferred, drawing on business-related factors where possible
- The implications of adopting the principle should also be identified
- Base principles on experience (graphical history, literature, etc.) to repeat what worked and avoid what did not work
- For each quality goal, consider whether there is a principle that will guide structuring decisions to achieve the goal

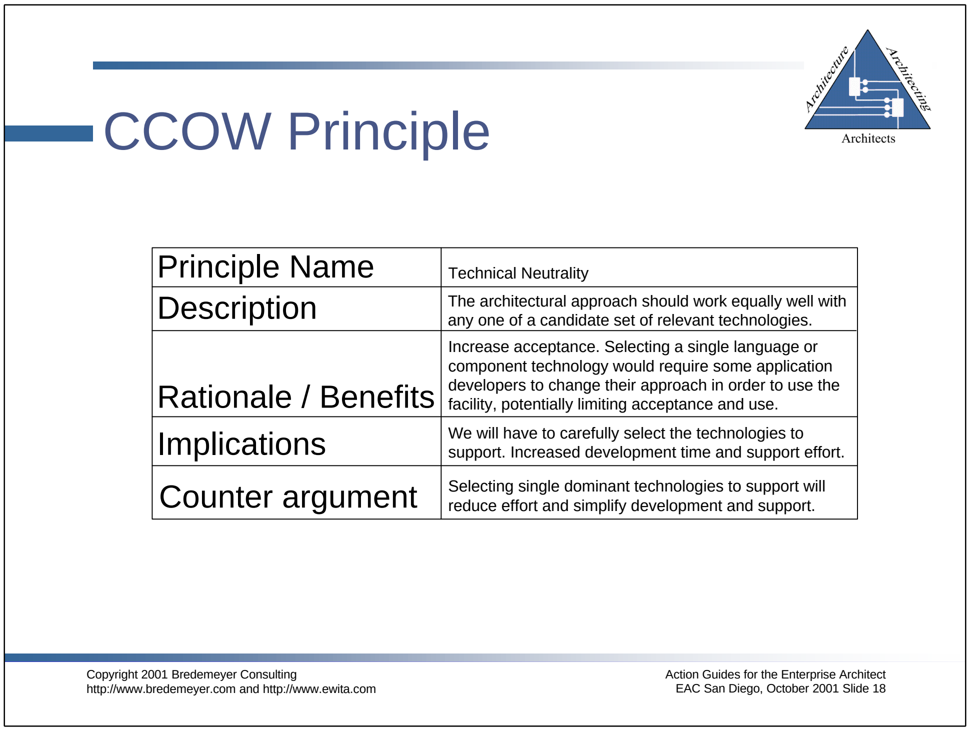
- Ref: Tapscott and Caston, *Paradigm Shift*, 1995

### Uses of Principles

Architectural principles

- help establish a context for architectural decisions by using business criteria to rationalize basic architectural choices, and
- eliminate the need for "evaluating endless alternatives in the modeling stages by agreeing up front on preferred directions." Tapscott and Caston, 1995 p.241

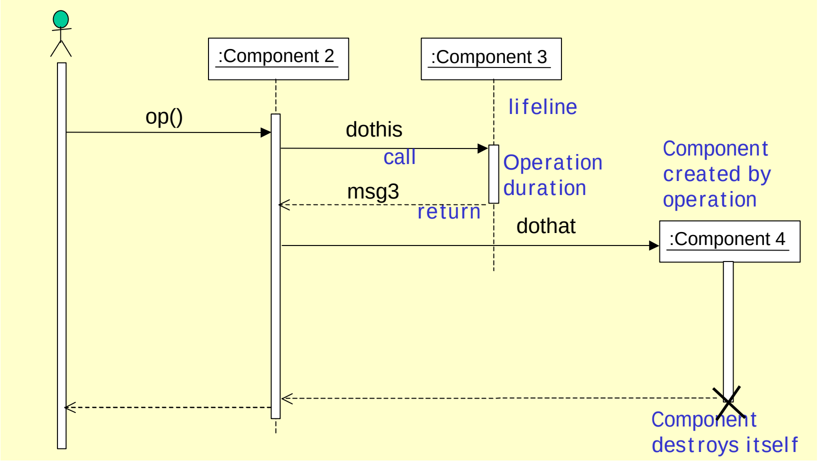
System Structuring Action Guide #2: Architectural Principles



# CCOW Principle

| | |
|---|---|
| Principle Name | Technical Neutrality |
| Description | The architectural approach should work equally well with any one of a candidate set of relevant technologies. |
| Rationale / Benefits | Increase acceptance. Selecting a single language or component technology would require some application developers to change their approach in order to use the facility, potentially limiting acceptance and use. |
| Implications | We will have to carefully select the technologies to support. Increased development time and support effort. |
| Counter argument | Selecting single dominant technologies to support will reduce effort and simplify development and support. |

## Sequence Diagram



**Architecture Action Guides from Bredemeyer Consulting**
**http://www.bredemeyer.com**

## Sequence Diagram

### Sequence Diagrams

- Sequence Diagrams show how components interact to accomplish some piece of system functionality.

### Sequence Diagram Notation

We suggest using the UML Sequence Diagram notation:

- Each component is shown as a rectangle in a separate column. A dashed line is drawn down from the component (to the point it is destroyed, if that happens in the sequence depicted). This line is called the lifeline.
- The vertical dimension represents time (generally time proceeds down the page). The horizon dimension represents objects (in our case, logical component instances) participating in an interaction.
- Each message is shown as a horizontal arrow from the lifeline of the component that sent the message to the component that receives it. The UML syntax for messages is as follows

  return := message(parameter : parameterType): returnType
- Arrows indicate

  ───▶ procedure call or other nested flow of control.

  ───> Flat flow of control.

  ───⇀ Asynchronous flow of control

  - - - -▶ Return from a procedure call

- Ref: Latest UML Specification, see http://www.rational.com/uml/index.jtmpl

### Uses of Sequence Diagrams

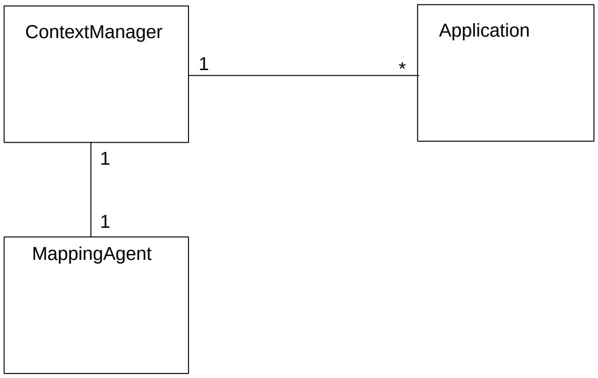
To the architecture team:

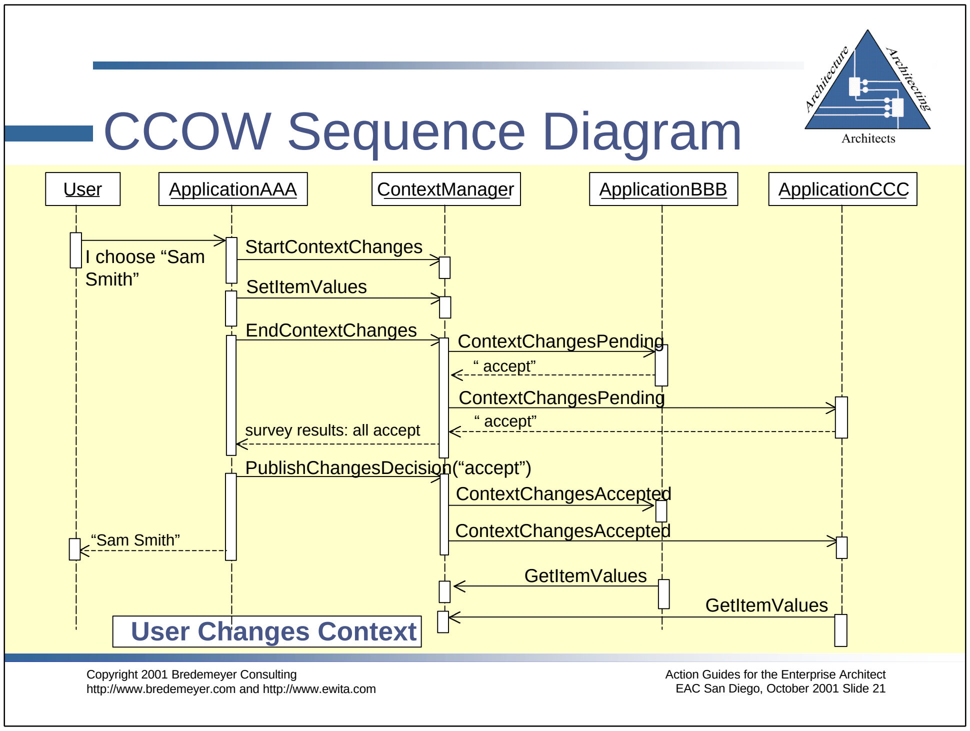
- helpful in reasoning about system behavior
- helpful in designing interfaces:
  - thinking through the allocation of responsibilities to components (what message is sent to what component),
  - the sequence of messages (interface semantics),
  - the details of the message syntax (operation specification on the interface)
- useful for back-of-the-envelope calculations of system qualities like performance, or reasoning about how system qualities like security will be met

To others:

- communicates/documents decisions about the behavior of the system at the architectural level

# CCOW Architecture Diagram





## CCOW Sequence Diagram

User

ApplicationAAA

ContextManager

ApplicationBBB

ApplicationCCC

I choose “Sam Smith”

StartContextChanges

SetItemValues

EndContextChanges

ContextChangesPending

“ accept”

ContextChangesPending

“ accept”

survey results: all accept

PublishChangesDecision(“accept”)

ContextChangesAccepted

ContextChangesAccepted

“Sam Smith”

GetItemValues

GetItemValues

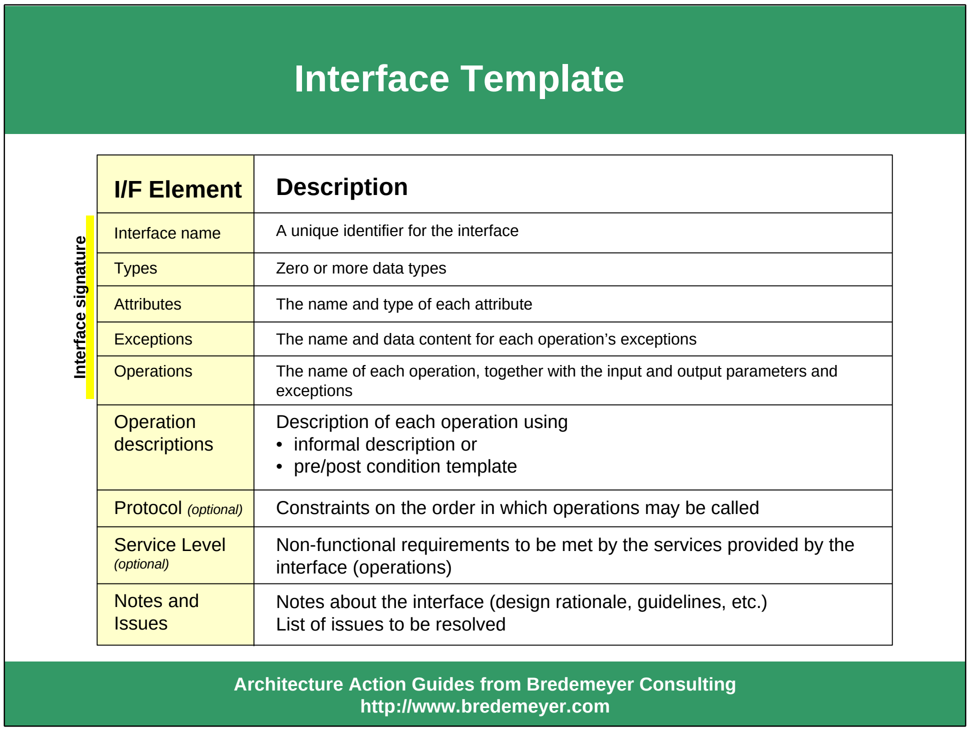
User Changes Context

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 21

## Assumptions

All applications have joined the common context by sending a message JoinCommonContext(applID, surveyYes) to the ContextManager.

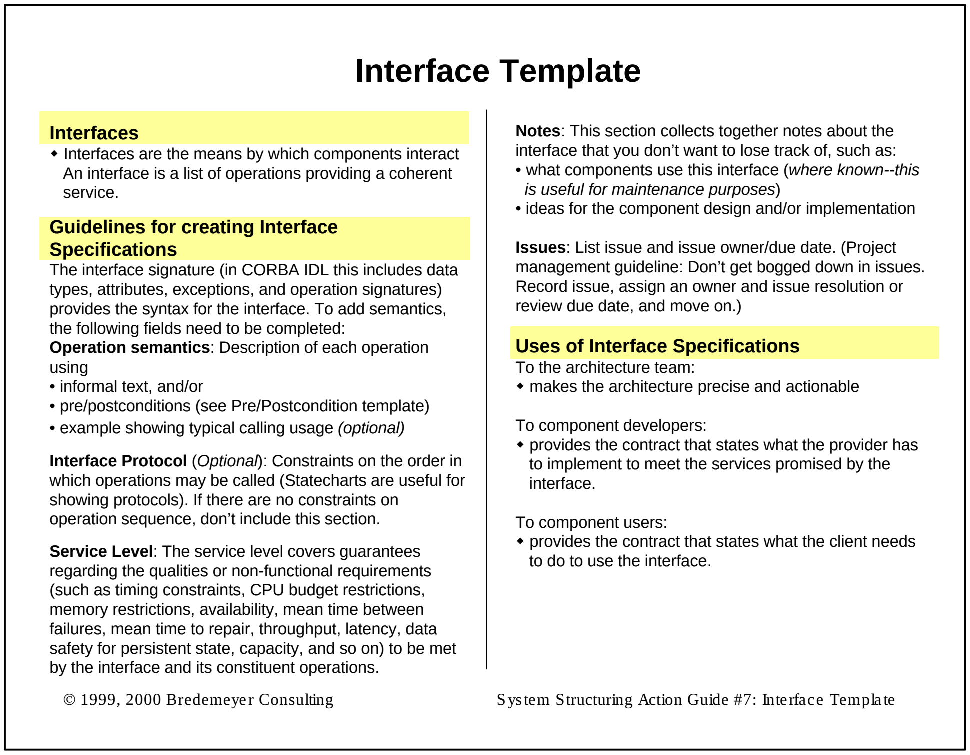


# Interface Template

Interface signature

| I/F Element | Description |
|---|---|
| Interface name | A unique identifier for the interface |
| Types | Zero or more data types |
| Attributes | The name and type of each attribute |
| Exceptions | The name and data content for each operation's exceptions |
| Operations | The name of each operation, together with the input and output parameters and exceptions |
| Operation descriptions | Description of each operation using<br>• informal description or<br>• pre/post condition template |
| Protocol *(optional)* | Constraints on the order in which operations may be called |
| Service Level *(optional)* | Non-functional requirements to be met by the services provided by the interface (operations) |
| Notes and Issues | Notes about the interface (design rationale, guidelines, etc.)<br>List of issues to be resolved |

**Architecture Action Guides from Bredemeyer Consulting**
**http://www.bredemeyer.com**



# Interface Template

## Interfaces

- Interfaces are the means by which components interact An interface is a list of operations providing a coherent service.

## Guidelines for creating Interface Specifications

The interface signature (in CORBA IDL this includes data types, attributes, exceptions, and operation signatures) provides the syntax for the interface. To add semantics, the following fields need to be completed:
**Operation semantics**: Description of each operation using
- informal text, and/or
- pre/postconditions (see Pre/Postcondition template)
- example showing typical calling usage *(optional)*

**Interface Protocol** (*Optional*): Constraints on the order in which operations may be called (Statecharts are useful for showing protocols). If there are no constraints on operation sequence, don't include this section.

**Service Level**: The service level covers guarantees regarding the qualities or non-functional requirements (such as timing constraints, CPU budget restrictions, memory restrictions, availability, mean time between failures, mean time to repair, throughput, latency, data safety for persistent state, capacity, and so on) to be met by the interface and its constituent operations.

**Notes**: This section collects together notes about the interface that you don't want to lose track of, such as:
- what components use this interface (*where known--this is useful for maintenance purposes*)
- ideas for the component design and/or implementation

**Issues**: List issue and issue owner/due date. (Project management guideline: Don't get bogged down in issues. Record issue, assign an owner and issue resolution or review due date, and move on.)

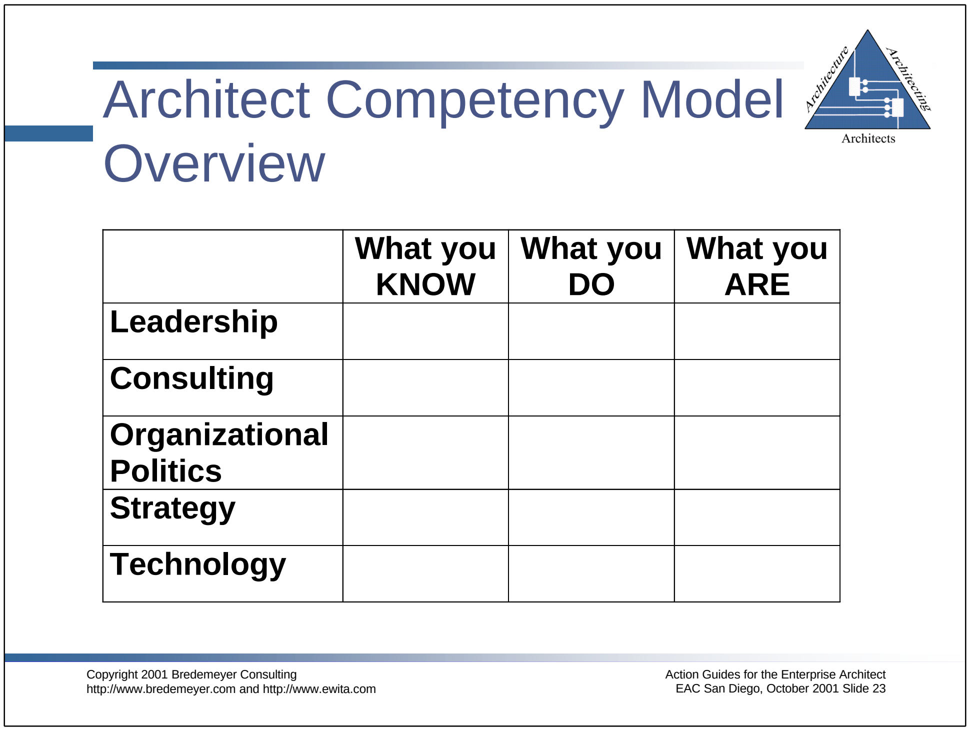
## Uses of Interface Specifications

To the architecture team:
- makes the architecture precise and actionable

To component developers:
- provides the contract that states what the provider has to implement to meet the services promised by the interface.

To component users:
- provides the contract that states what the client needs to do to use the interface.

System Structuring Action Guide #7: Interface Template



## Architect Competency Model Overview

| | What you KNOW | What you DO | What you ARE |
|---|---|---|---|
| Leadership | | | |
| Consulting | | | |
| Organizational Politics | | | |
| Strategy | | | |
| Technology | | | |

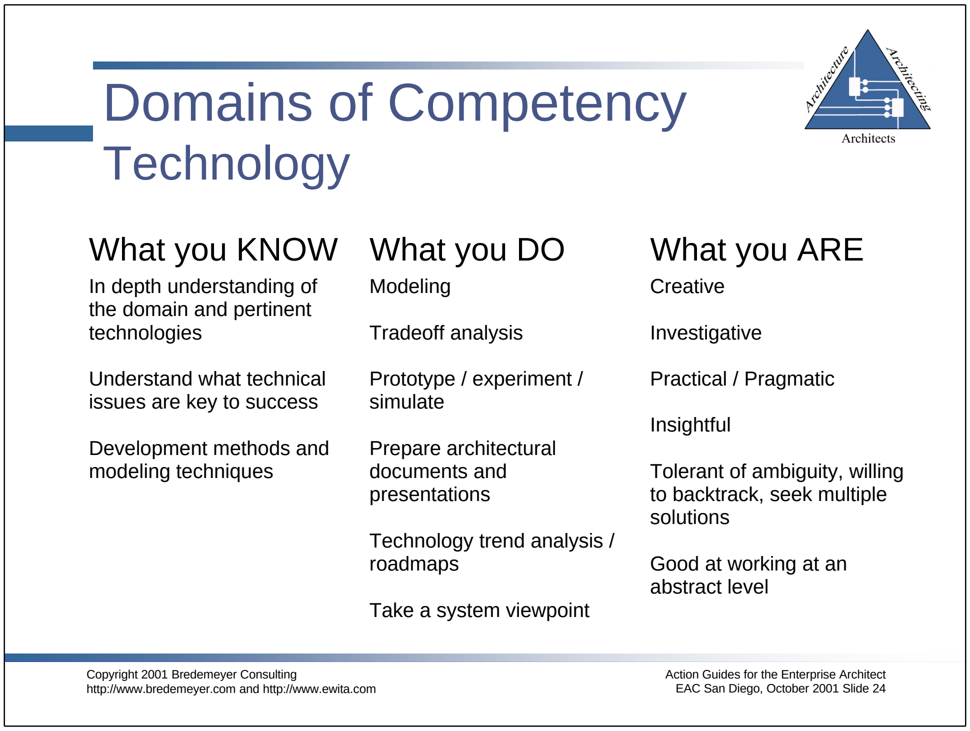
### “5 Hats of the Architect”

This model shows the 5 primary roles played by the architect. At different points in the process, different roles will be more foreground and others will be more background. Also, the lead architect will tend to take on more of the strategy, politics and leadership responsibilities than the junior architects on the team. This, in turn, means that the lead architect has to have more of a predilection for, and talents and skills in, those areas.

For each of the 5 roles or domains of competency, the model answers:

- What knowledge does an architect need in this area?
- What activities does an architect do in this area?
- What predilections, proficiencies and talents does a good architect have in this area?

This model is fully described in our paper titled “The Role of the Architect” available at http://www.ewita.com/newsletters/10021%Files/ArchitectRole.PDF.



## Domains of Competency Technology

### What you KNOW

In depth understanding of the domain and pertinent technologies

Understand what technical issues are key to success

Development methods and modeling techniques

### What you DO

Modeling

Tradeoff analysis

Prototype / experiment / simulate

Prepare architectural documents and presentations

Technology trend analysis / roadmaps

Take a system viewpoint

### What you ARE

Creative

Investigative

Practical / Pragmatic

Insightful

Tolerant of ambiguity, willing to backtrack, seek multiple solutions

Good at working at an abstract level

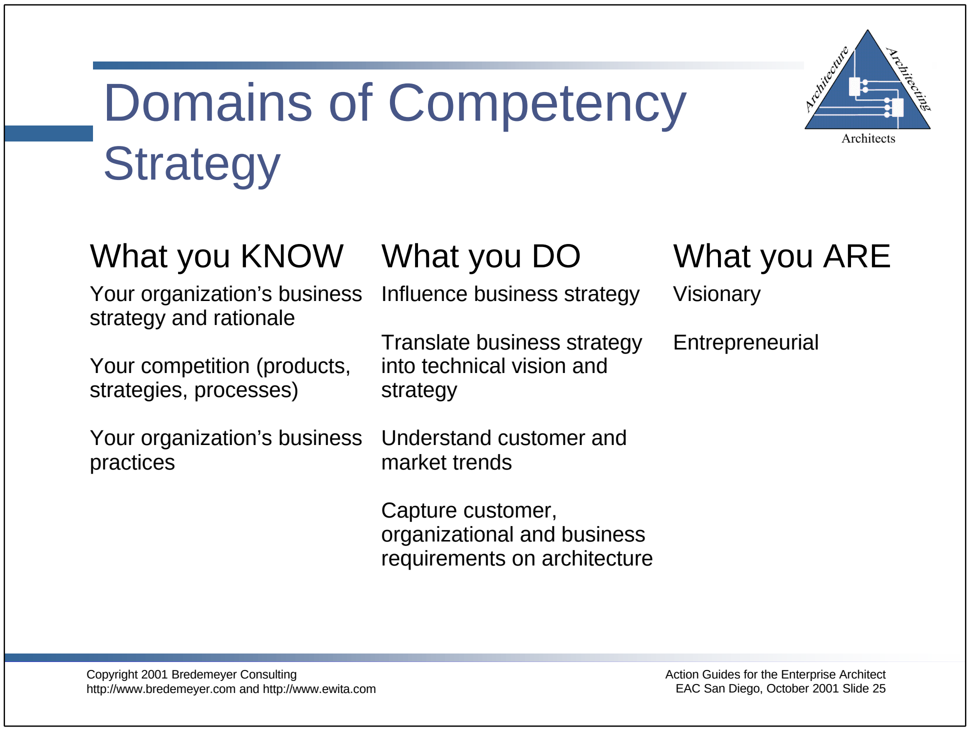
## Technology

As an architect, you need a thorough knowledge of your organization's product domain, relevant technologies and development processes. But even in the technical area, your key activities are different than those of the developers. The problems are less well-defined, often with unclear or conflicting objectives, and you play a significant role in clarifying what the objectives are. Your focus is more on the implications of organizational objectives on technical choices. You take an overall system view. You are building models of the problem and solution space, exploring alternative approaches, preparing documents and explaining the architecture to sponsors and stakeholders.

The personal characteristics really essential to success in this domain are a high tolerance for ambiguity and a lot of skill working consistently at an abstract level. We know of at least one case where an otherwise qualified junior architect did not get the senior architect position because of his need for clear and unambiguous objectives.

Often this is the extent of how people see the architect role, and this, along with consulting, is in fact the primary role of a junior architect. But as a senior architect you also need to be an effective strategist.

If the junior architect is primarily a technologist, the senior architect is primarily a strategist, contributing to the business strategy and having primary responsibility for defining the technical strategy.



## Domains of Competency
## Strategy

**What you KNOW**

Your organization's business strategy and rationale

Your competition (products, strategies, processes)

Your organization's business practices

**What you DO**

Influence business strategy

Translate business strategy into technical vision and strategy

Understand customer and market trends

Capture customer, organizational and business requirements on architecture

**What you ARE**

Visionary

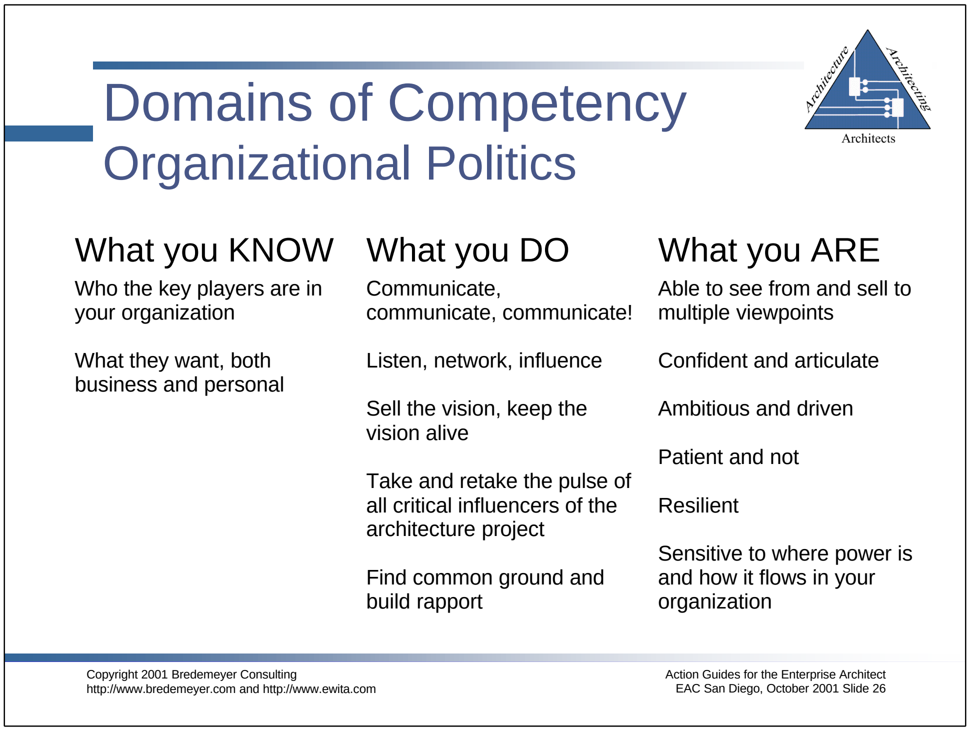
Entrepreneurial

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 25

### Strategy

To succeed in this aspect of the architect role, you need a solid understanding of your organization's business strategy and the rationale behind it, as well as your company or division's business practices, planning cycles, and decision making processes. You have a good understanding of the business context of your organization. You understand your competitors, their products, strategies and product generation processes. You are familiar with the key factors in the business environment that affect your organization's success, and you are able to distill all these business factors into architectural requirements and architectural choices. But the overriding characteristic that fuels your success in this domain is that of an entrepreneurial visionary who can translate well between the business and technical domains.

As a skilled technologist you create good architecture. As a skilled strategist, you create the right architecture for your organization.  The next three domains of competency are more about getting the architecture realized. The first is about gaining support for the architecture among the management community. Rob Seliger, the principal architect for the Concert Architecture (Seliger, 1997) for medical information systems said, the single thing architects most need to learn is how to sell, sell, sell.



# Domains of Competency
## Organizational Politics

### What you KNOW

Who the key players are in your organization

What they want, both business and personal

### What you DO

Communicate, communicate, communicate!

Listen, network, influence

Sell the vision, keep the vision alive

Take and retake the pulse of all critical influencers of the architecture project

Find common ground and build rapport

### What you ARE

Able to see from and sell to multiple viewpoints

Confident and articulate

Ambitious and driven

Patient and not

Resilient

Sensitive to where power is and how it flows in your organization

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 26

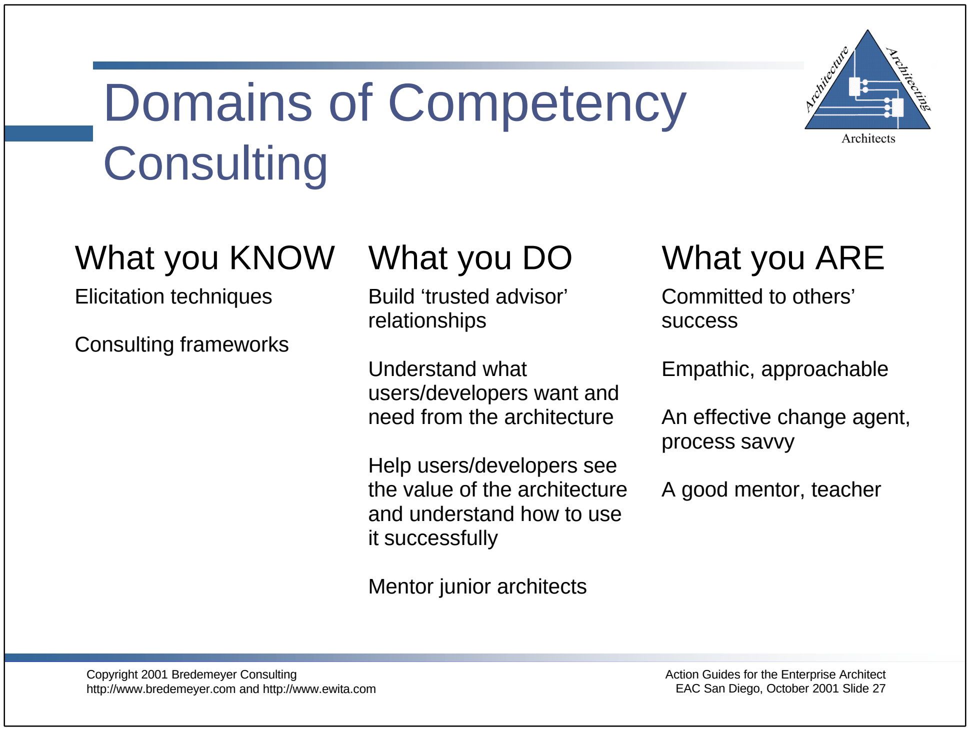
## Organizational Politics

Architectures almost always have many and diverse stakeholders, and are ultimately meant to be used by many developers. Often they are used across divisions and by developers in other companies. To gain and maintain the sponsorship of your management and the enthusiastic support of other key influencers, you will need to do a good deal of influencing yourself.

You really need to understand both the business and personal objectives of key players, and get them personally committed to the success of the architecture. This means listening, networking, articulating and selling a vision, and doing all this continuously over the life of the project.

The people doing this well are extremely articulate and confident. They are resilient and driven, and they are sensitive to where the real power is and how it flows. They look for and see the organization behind the organization, and they use this insight to build and maintain support for their projects.

This domain of competency generates the organizational support to get the architecture created. The next one supports getting it deployed into use.



## Domains of Competency
## Consulting

### What you KNOW

Elicitation techniques

Consulting frameworks

### What you DO

Build 'trusted advisor' relationships

Understand what users/developers want and need from the architecture

Help users/developers see the value of the architecture and understand how to use it successfully

Mentor junior architects

### What you ARE

Committed to others' success

Empathic, approachable

An effective change agent, process savvy

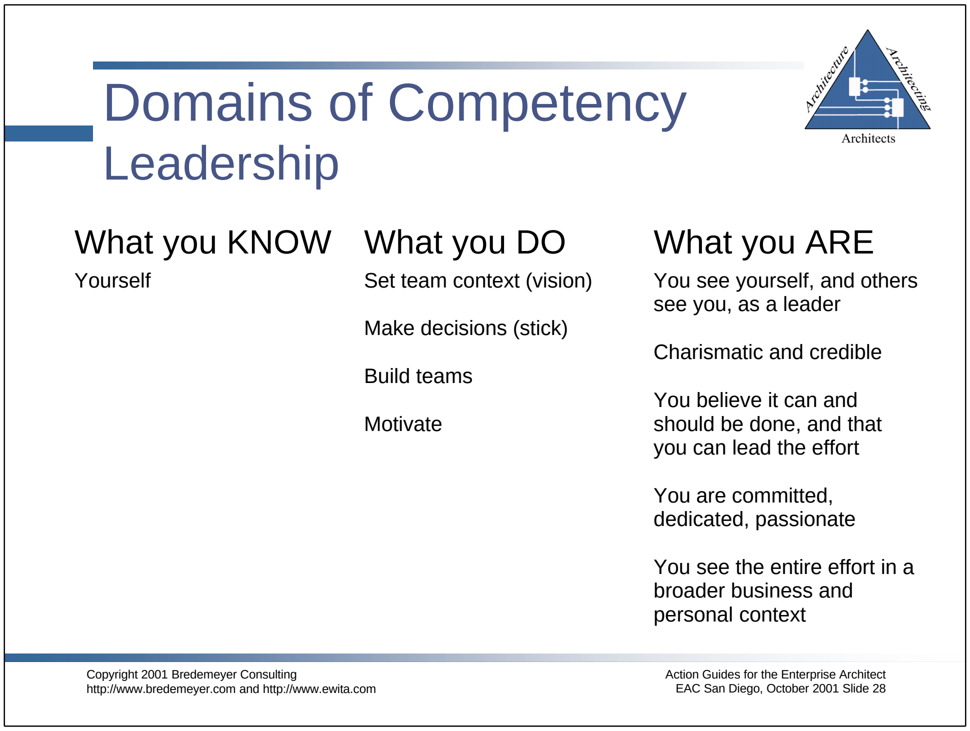
A good mentor, teacher

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 27

## Consulting

The actual users of architecture are development teams creating products or components, and their goal is not to make your architecture successful, but rather to satisfy their specific functionality, schedule and quality requirements. While using the architecture may be the best overall approach for the organization, this is often not apparent to its users. Consequently, your task as an architect includes recognizing first that developers are a primary customer, and that the architecture must provide value to them in generating good products. Second, you need to enable development teams to quickly understand and effectively use the architecture. You are functioning here more as a mentor and teacher, preparing and making presentations, consulting to individuals and teams, and also mentoring junior architects.

What really contributes to your success here is to be truly committed to others' success and to have a good understanding of change management and how groups adopt new processes.



# Domains of Competency
# Leadership

### What you KNOW

Yourself

### What you DO

Set team context (vision)

Make decisions (stick)

Build teams

Motivate

### What you ARE

You see yourself, and others see you, as a leader

Charismatic and credible

You believe it can and should be done, and that you can lead the effort

You are committed, dedicated, passionate

You see the entire effort in a broader business and personal context

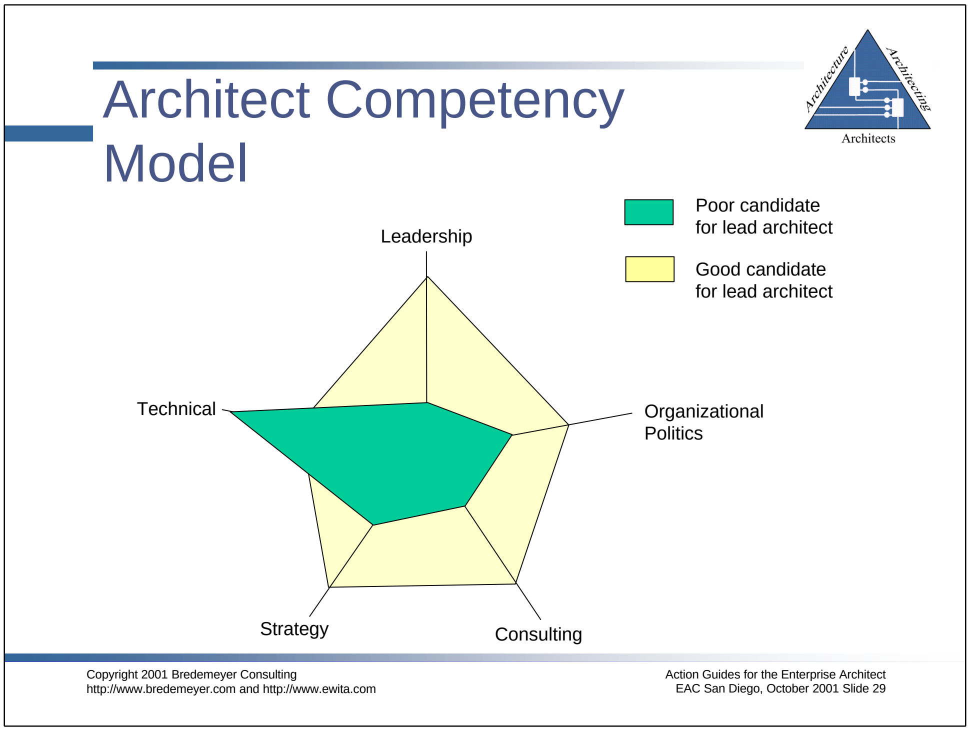
## Leadership

The domain of competency which organizes all the others and gives them dynamic force, is leadership. An architecture team without leadership goes nowhere. It thrashes and diverges. We've seen this too many times. A leader is required to infuse the team with a common vision, and to motivate the core team and associated teams to do their best work.

This requires dedication and passion, a strong belief that you can lead the effort and the desire to do so (Nigel Nicholson writes "desire to lead is perhaps the most important characteristic a leader can possess.", HBR July-Aug 1998, p. 142). You must see yourself, and others must see you, as a credible leader.

## What makes a Leader?

Daniel Goleman's Harvard Business Review article of this title ("What makes a Leader?" Nov-Dec 1998) identifies intellect and cognitive skills like big-picture thinking and long-term vision as important drivers of outstanding performance among leaders--but emotional intelligence outscored technical skills and IQ, in importance to success as a leader.  See p 95 for a useful summary of the five components of emotional intelligence identified by Goleman. Note that many of the "hallmarks of leaders" identified by Goleman map onto different facets of our model--e.g., "social skill" maps onto organizational politics.



## Architect Competency Model

🟩 Poor candidate for lead architect

🟨 Good candidate for lead architect

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

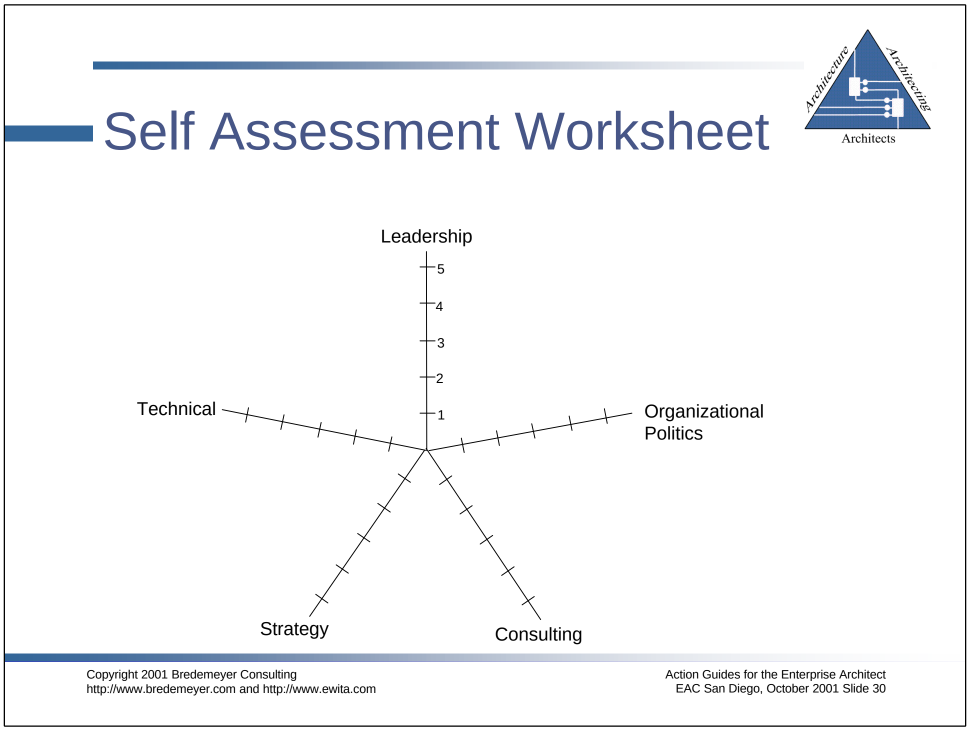
Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 29

### To Be or Not to Be

This diagram is known as a Kiviat or spider diagram. In this Kiviat for the Role of the Architect, a fairly well filled out profile indicates that you are well-suited to the job of lead architect.  (See page 27 for worksheet).

If your profile is skewed towards the technical, without good leadership and social skills, you would do better to let someone else lead the architecture effort while you focus on technical concerns. This is not to say you should not be on the architecting team--except if you have a large ego around your technical opinions and think you always “know better” than everyone else. In this case, you’re not a good follower and should probably steer clear of the architecture work if you don’t want to slow it down to the point of failure. (*end of soapbox* ☺)

Many teams solve the problem of finding all the skills in one person by splitting the leadership responsibilities across a team manager and lead architect.



# Self Assessment Worksheet

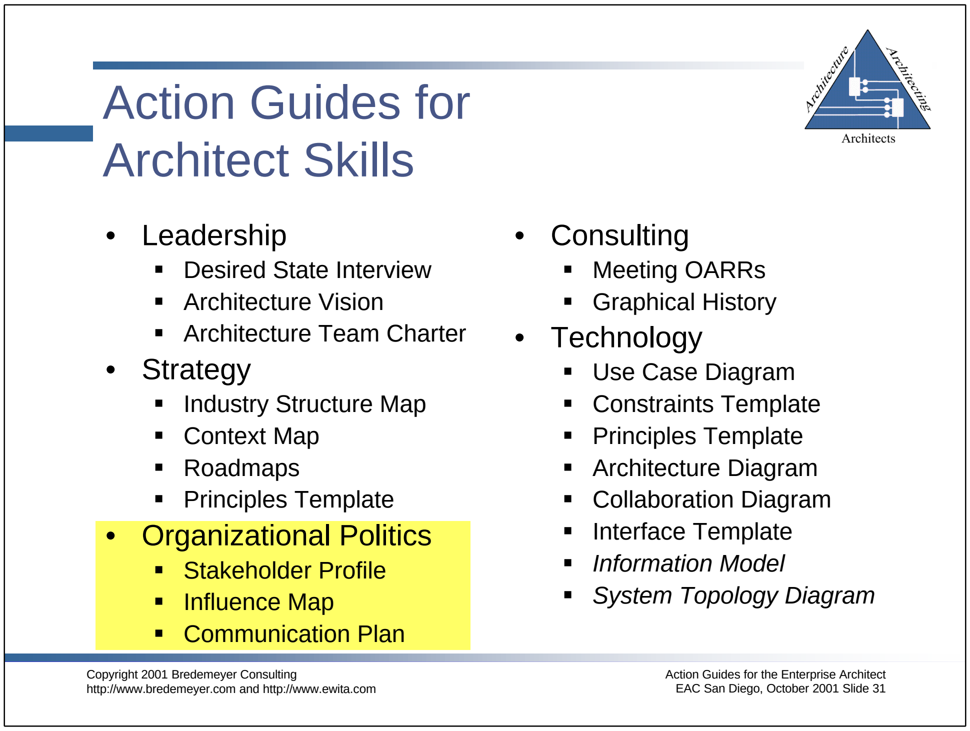
Leadership
5
4
3
2
1
Technical
Organizational Politics
Strategy
Consulting

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 30

## Self Assessment

Rate yourself on a scale of 1 (weak) to 5 (strong), for each of the domains of competency and plot on the chart above. Use this to help you direct your own personal development attention.



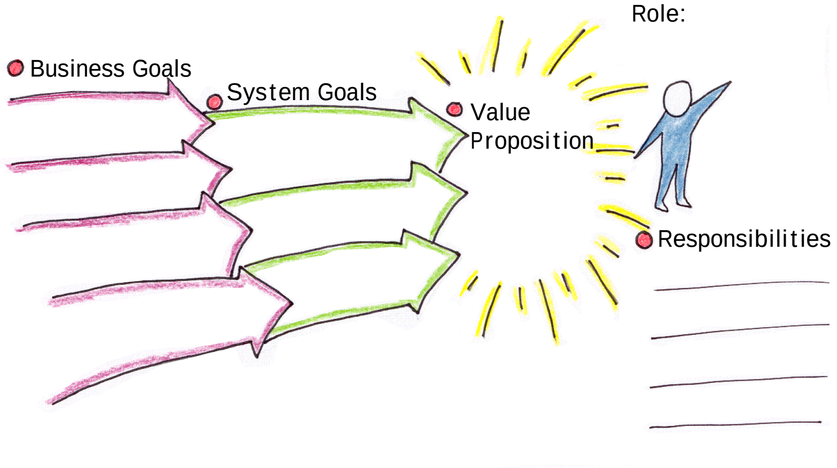
# Action Guides for Architect Skills

- Leadership
  - Desired State Interview
  - Architecture Vision
  - Architecture Team Charter
- Strategy
  - Industry Structure Map
  - Context Map
  - Roadmaps
  - Principles Template
- Organizational Politics
  - Stakeholder Profile
  - Influence Map
  - Communication Plan
- Consulting
  - Meeting OARRs
  - Graphical History
- Technology
  - Use Case Diagram
  - Constraints Template
  - Principles Template
  - Architecture Diagram
  - Collaboration Diagram
  - Interface Template
  - *Information Model*
  - *System Topology Diagram*

http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 31

# Stakeholder Profile



Role:

Business Goals

System Goals

Value Proposition

Responsibilities

Architecture Action Guides from Bredemeyer Consulting
http://www.bredemeyer.com

# Stakeholder Profile

### Purpose of the Stakeholder Profile Template

- Record stakeholder responsibilities and related goals, so that the requirements gathering and system structuring activities can be focused on the most valuable areas from the perspective of important stakeholders.
- Provide an effective graphical group communication medium.

### Guidelines for creating a Stakeholder Profile

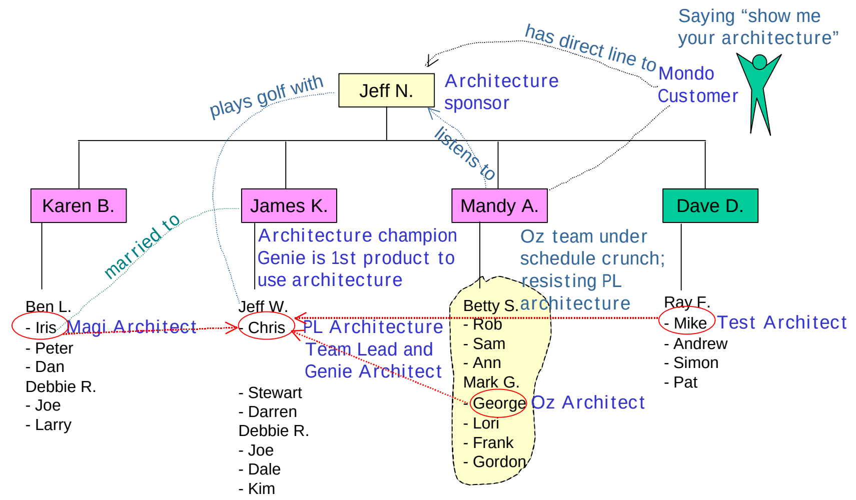
For each substantive stakeholder role, create a stakeholder profile. Interview stakeholders, using the template to capture responsibilities and goal information

1. List the responsibilities associated with the role, that are relevant to the system under design
2. List the goals associated with major tasks the stakeholder has related to accomplishing his/her tasks most effectively. Pay attention not just to goals related to functionality but also qualities of the system (e.g., performance). Watch for goals disguised as concerns (e.g., the last system was too slow can be translated into a goal for the new system).
3. Ask questions about critical success factors, deficiencies in past systems, as well as opportunities and needs.

### Uses of Stakeholder Goal Information

- Used to scope the architecture and the systems it supports.
- Drive creation of principles, selection of architecture styles, and selection/creation of mechanisms in the meta-architecture phase (Structuring Pass 1).
- Functionality goals are the starting point for use cases (i.e., functional requirements specification) (Architectural Requirements Pass 2)
- Quality goals are the starting point for quality (i.e., non-functional) requirements specification (Architectural Requirements Pass 2).

# Architecture Influence Map



Saying “show me your architecture”

has direct line to

Mondo Customer

plays golf with

Jeff N.

Architecture sponsor

listens to

Karen B.

James K.

Mandy A.

Dave D.

married to

Architecture champion
Genie is 1st product to
use architecture

Oz team under
schedule crunch;
resisting PL
architecture

Ben L.
- Iris Magi Architect
- Peter
- Dan
Debbie R.
- Joe
- Larry

Jeff W.
- Chris PL Architecture Team Lead and Genie Architect
- Stewart
- Darren
Debbie R.
- Joe
- Dale
- Kim

Betty S.
- Rob
- Sam
- Ann
Mark G.
- George Oz Architect
- Lori
- Frank
- Gordon

Ray F.
- Mike Test Architect
- Andrew
- Simon
- Pat

# Architecture Influence Map

## Architecture Influence Map

- Influence maps reflect the informal paths of influence in the organization. These paths may be through dotted-line relationships, informal teams, and other informal work and personal relationships.

## Guidelines for Creating an Influence Map

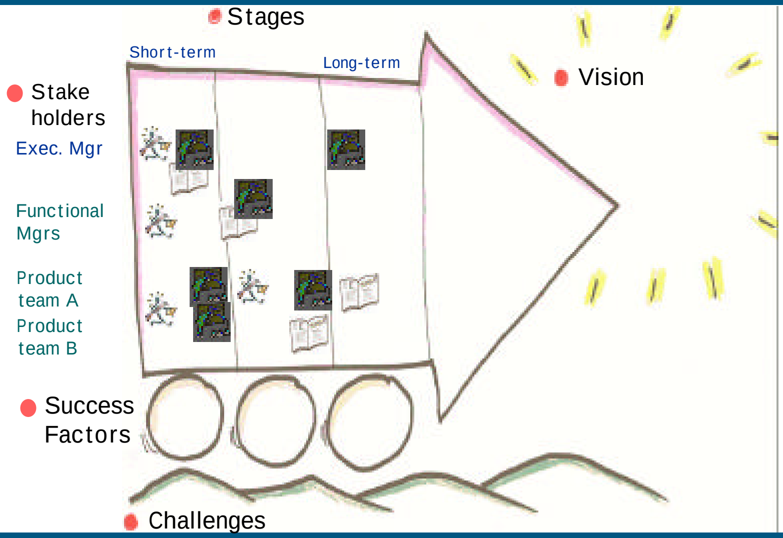
As relevant to the architecting effort,

- identify the informal structures and relationships (like dotted-line reporting relationships) on the annotated Org. Chart
- Also think about who influences the architecture directly? Who influences the architecture indirectly? Who influences the success or impedes the architecture directly? Indirectly? Who, among the key players, is a supporter? Opponent? Neutral?
- Do these paths of influence map onto the formal structure, or reflect informal networks of influence that you need to be aware of? Are they counterproductive or conducive to the architecture effort?
- Think about alternative scenarios. What if Chris leads the architecture team? Or George? What different political forces will they have to contend with? What paths of influence support one or the other, and which would tend to work against them?

## Uses of Influence Map Information

- An Influence Map is a powerful tool, as it identifies communication hot spots and highlights where to focus effort in gaining sponsorship for, and commitment to, the architecture.
- Use the Influence Map in designing your Communication Plan.

## Architecture Communication Plan



● Stages

Short-term

Long-term

● Vision

● Stake holders

Exec. Mgr

Functional Mgrs

Product team A

Product team B

● Success Factors

● Challenges

Architecture Action Guides from Bredemeyer Consulting
http://www.bredemeyer.com

## Architecture Communication Plan

### Purpose of the Communication Plan Template

- Facilitate planning what input to gather and what to communicate, to whom, in what format.
- Get agreement on phases and deliverables for the architecting project.
- Provide an effective graphical group communication medium, and build alignment around a common communication plan. Boost team performance.

### Guidelines for creating a Communication Plan

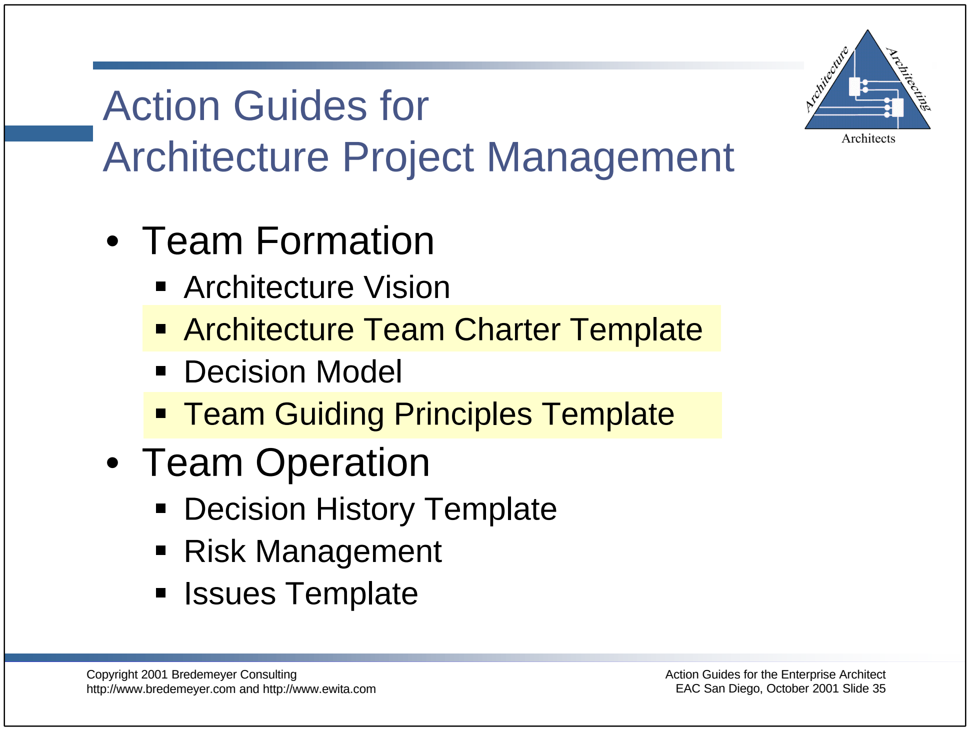
1. *Identify architecture stakeholders* and their communication needs, styles and timing.
2. *Identify the Phases* and map over time
3. *Decide on graphic icons* for communication form (presentation, tutorial, document, etc.) and whether the communication is inward-bound to the team, or outward bound to others in the organization.
3. *Map the communication over time:* For each phases, identify what needs to be communicated to whom, and in what form
4. *Identify risks over time*

### Uses of Communication Plan Information

- Orient the team towards communicating.
- Build communication into the project schedule, to help ensure it happens.
- Help architect the documentation, so that it is most helpful to the various stakeholders, and to you in getting their buy-in, feedback, and adherence.

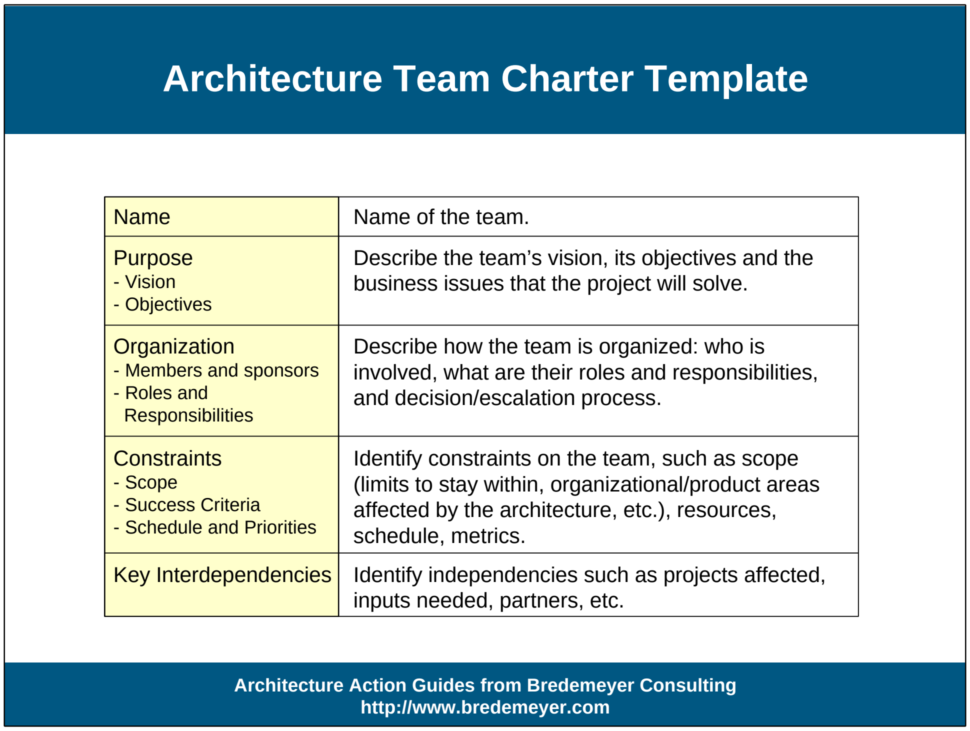
❋ Adapted from Grove Consultants International Graphic Guide #12 (1-800-49GROVE)

Init/Commit Action Guide #2: Communication Plan



## Action Guides for Architecture Project Management

- Team Formation
  - Architecture Vision
  - Architecture Team Charter Template
  - Decision Model
  - Team Guiding Principles Template
- Team Operation
  - Decision History Template
  - Risk Management
  - Issues Template



# Architecture Team Charter Template

| | |
|---|---|
| Name | Name of the team. |
| Purpose<br>- Vision<br>- Objectives | Describe the team’s vision, its objectives and the business issues that the project will solve. |
| Organization<br>- Members and sponsors<br>- Roles and Responsibilities | Describe how the team is organized: who is involved, what are their roles and responsibilities, and decision/escalation process. |
| Constraints<br>- Scope<br>- Success Criteria<br>- Schedule and Priorities | Identify constraints on the team, such as scope (limits to stay within, organizational/product areas affected by the architecture, etc.), resources, schedule, metrics. |
| Key Interdependencies | Identify independencies such as projects affected, inputs needed, partners, etc. |

Architecture Action Guides from Bredemeyer Consulting
http://www.bredemeyer.com

# Architecture Team Charter Template

### Purpose of the Team Charter

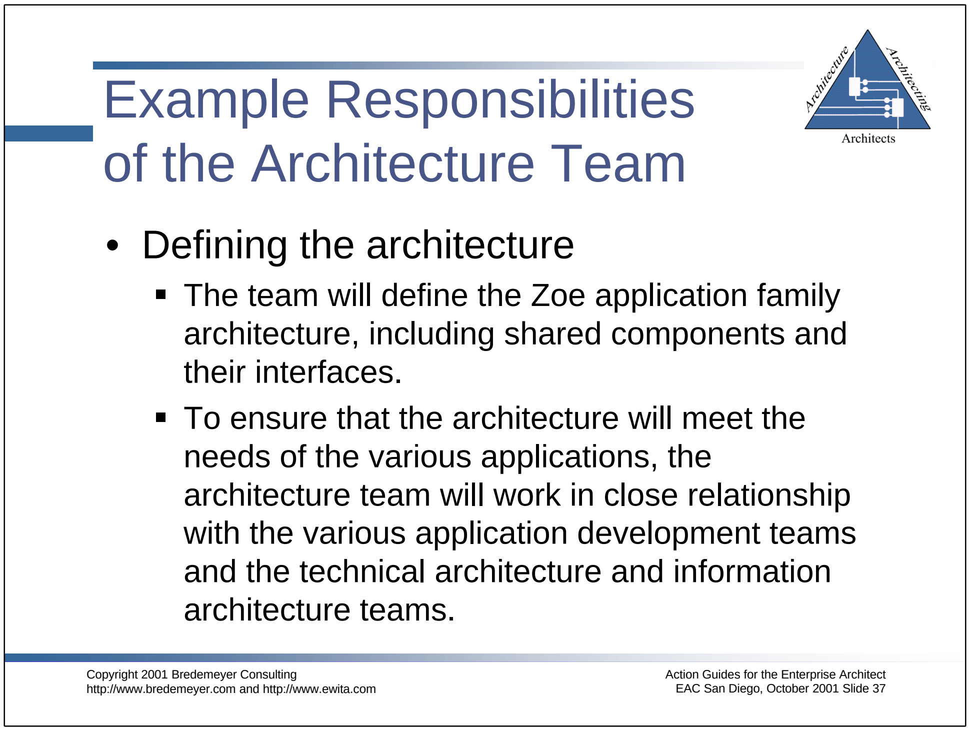
- The team charter serves to clarify to the team and others what the team’s purpose and responsibilities are.
- It defines the goals of the team, its scope/boundaries, and its responsibilities and extent of authority

### Guidelines for Creating a Team Charter

- ***Purpose.*** State the team’s vision or desired state to be achieved with the architecture. State the objectives of the architecture, and link to business strategy, objectives and issues.
- ***Organization.*** Describe how the team is organized: who is involved, who the sponsor(s) are, what the reporting relationships are, what the roles and responsibilities of the team leader and members are, how decisions will be made, how issues are to be resolved, what the escalation process is. It is also very helpful to include what the team’s responsibilities exclude (an IS NOT responsible for section).
- ***Constraints.*** Lay out any constraints on the team, such as scope of the architecture effort (which products it will cover, what markets it will address, which organizations or projects will be affected), resource constraints, schedule constraints, etc.
- ***Key Interdependencies.*** Identify any projects that depend on the architecture team (e.g., projects that will first use the architecture, pilot projects, etc.), as well as projects or individuals that the architecture project relies on.

### Uses of Team Charter Information

- The team charter is used to inform others in the organization of the team’s charter and role.
- It helps to sell what the team is doing, to set expectations appropriately, and to avoid turf conflicts.



## Example Responsibilities of the Architecture Team

- Defining the architecture
  - The team will define the Zoe application family architecture, including shared components and their interfaces.
  - To ensure that the architecture will meet the needs of the various applications, the architecture team will work in close relationship with the various application development teams and the technical architecture and information architecture teams.

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 37

### Other Responsibilities

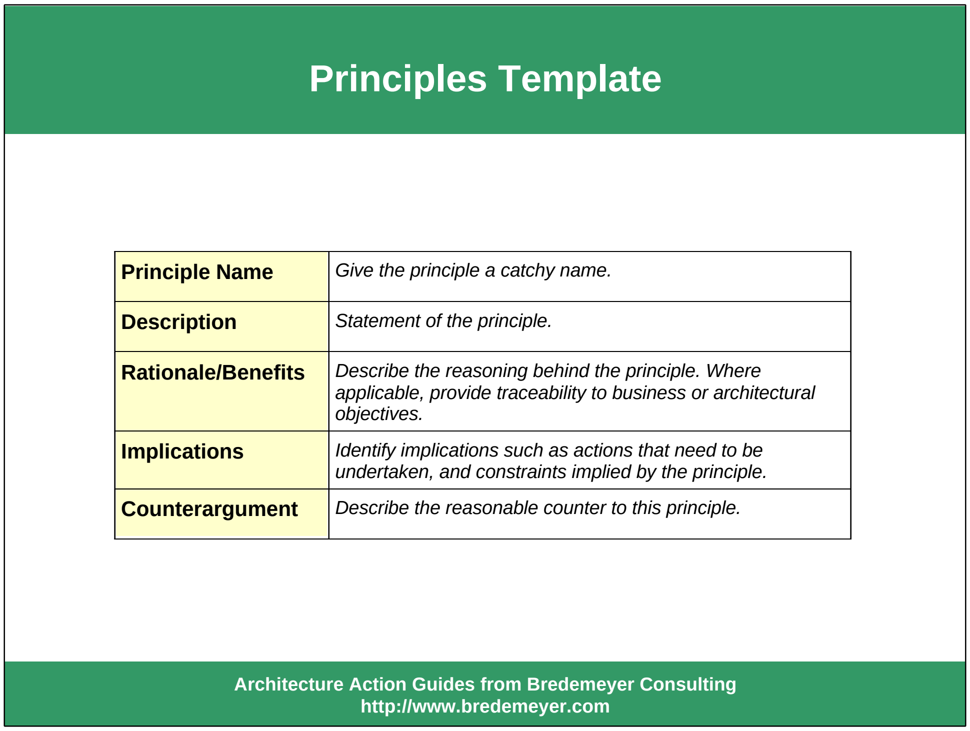
***Assessing Technical Risks***. The architecture team is responsible for maintaining a list of perceived technical software-related risks. They will experiment with critical technologies needed to make the architectural approach viable, before adopting it in the architecture.

***Develop Documentation and Help Prepare Training Material***. Develop detailed architecture documents and views, including architecture specifications and explanatory information for communicating to the engineering community, especially component owners. Help develop tutorials and demos to assist the development community in applying the architecture.

***Oversee the Architecture Rollout.*** The team will help develop and implement the architecture rollout plan. Members of the architecture team will provide assistance to development teams to help them use the architecture, and to resolve specific issues especially when they have systemic impact.

***Maintaining Architectural Integrity***. The architecture team must be notified of design/coding issues that impact the architecture. All changes to major interfaces and all explicit violations of the architecture must be approved by the architecture team. The architecture team is responsible for the development and maintenance of design and programming guidelines. The team is involved in the organization of design and code reviews to ensure that those guidelines are being followed.

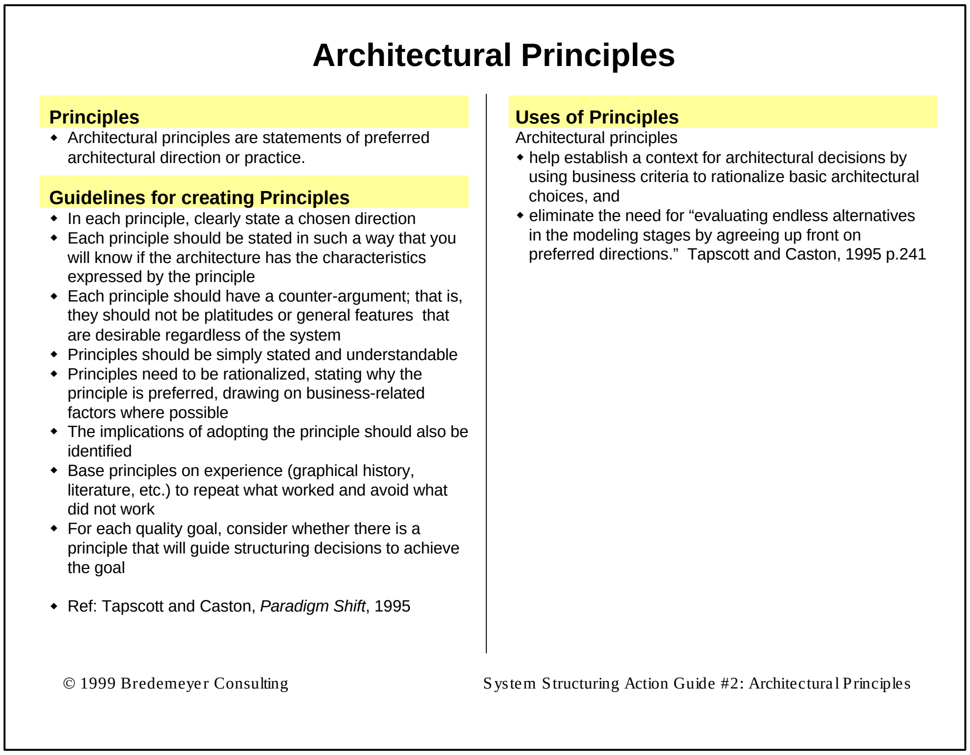
(We have adapted and added to Philippe Kruchten's Architecture Team Charter in his paper "The Architects: The Software Architecture Team" in *Software Architecture*, Patrick Donohoe (ed), 1999.)



## Principles Template

| | |
|---|---|
| **Principle Name** | *Give the principle a catchy name.* |
| **Description** | *Statement of the principle.* |
| **Rationale/Benefits** | *Describe the reasoning behind the principle. Where applicable, provide traceability to business or architectural objectives.* |
| **Implications** | *Identify implications such as actions that need to be undertaken, and constraints implied by the principle.* |
| **Counterargument** | *Describe the reasonable counter to this principle.* |

**Architecture Action Guides from Bredemeyer Consulting**
**http://www.bredemeyer.com**



## Architectural Principles

### Principles

- Architectural principles are statements of preferred architectural direction or practice.

### Guidelines for creating Principles

- In each principle, clearly state a chosen direction
- Each principle should be stated in such a way that you will know if the architecture has the characteristics expressed by the principle
- Each principle should have a counter-argument; that is, they should not be platitudes or general features that are desirable regardless of the system
- Principles should be simply stated and understandable
- Principles need to be rationalized, stating why the principle is preferred, drawing on business-related factors where possible
- The implications of adopting the principle should also be identified
- Base principles on experience (graphical history, literature, etc.) to repeat what worked and avoid what did not work
- For each quality goal, consider whether there is a principle that will guide structuring decisions to achieve the goal

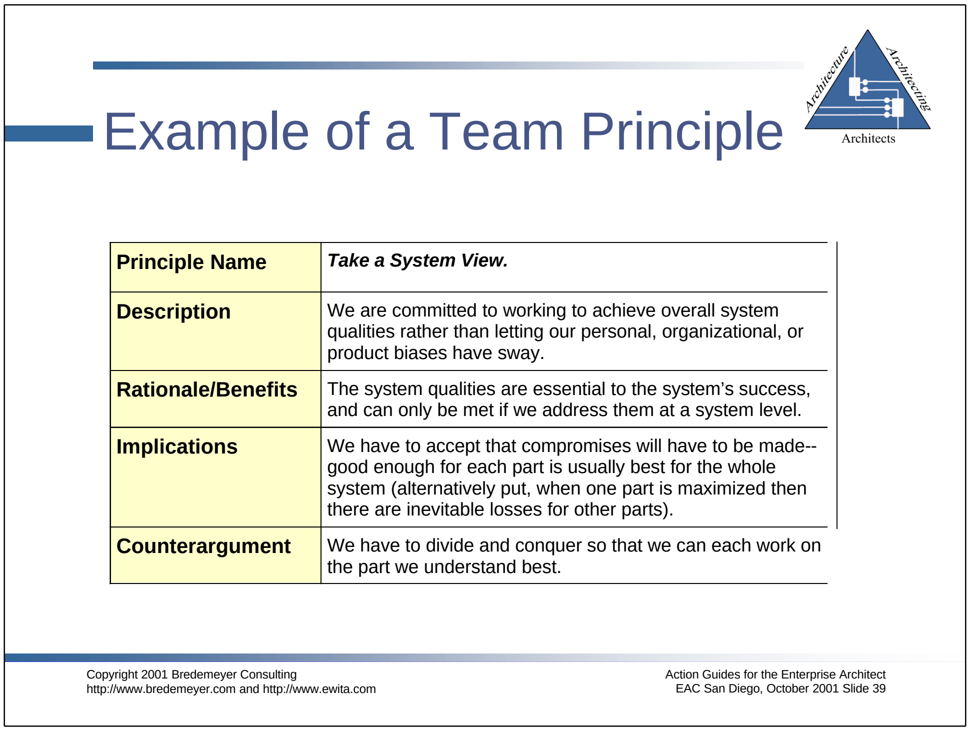
- Ref: Tapscott and Caston, *Paradigm Shift*, 1995

### Uses of Principles

Architectural principles

- help establish a context for architectural decisions by using business criteria to rationalize basic architectural choices, and
- eliminate the need for "evaluating endless alternatives in the modeling stages by agreeing up front on preferred directions." Tapscott and Caston, 1995 p.241

System Structuring Action Guide #2: Architectural Principles



# Example of a Team Principle

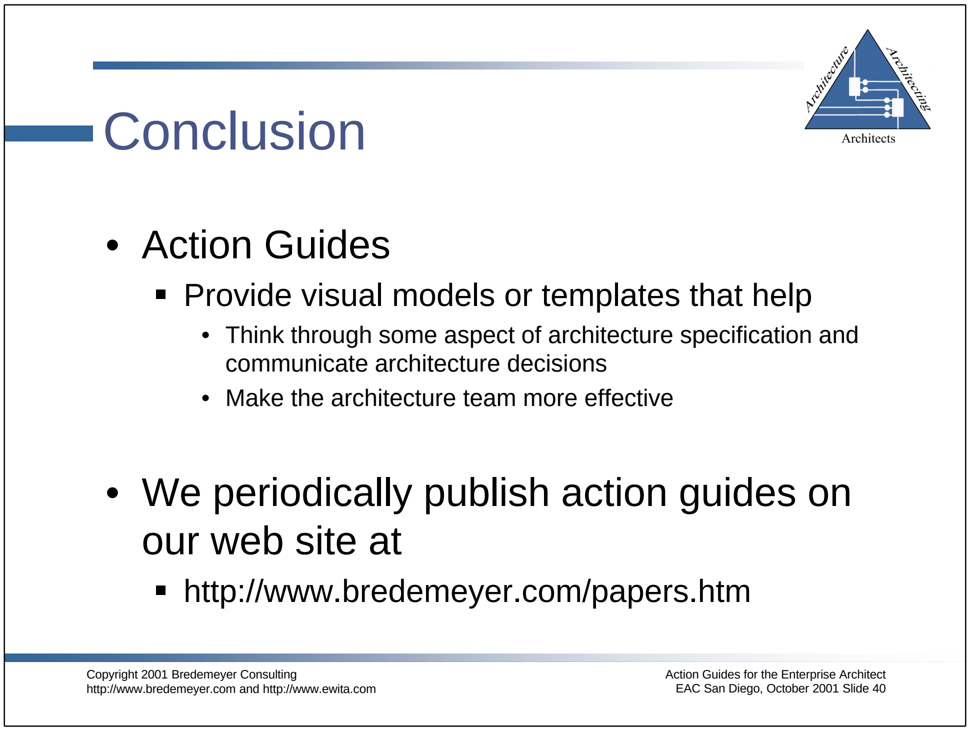
| Principle Name | ***Take a System View.*** |
|---|---|
| **Description** | We are committed to working to achieve overall system qualities rather than letting our personal, organizational, or product biases have sway. |
| **Rationale/Benefits** | The system qualities are essential to the system's success, and can only be met if we address them at a system level. |
| **Implications** | We have to accept that compromises will have to be made--good enough for each part is usually best for the whole system (alternatively put, when one part is maximized then there are inevitable losses for other parts). |
| **Counterargument** | We have to divide and conquer so that we can each work on the part we understand best. |

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 39

## Product Family Example

In the case of an architecture for a product family (a.k.a. product line), a similar "Take a System View" principle could emphasize the need to prioritize the product family over individual products in making architectural tradeoffs to meet the system qualities. This would give product architects on the team a strong admonition to consider the overall solution and accept some compromises to their product--some things will have to be given up at the product level, to create an architecture that is good enough for all products. It is well to remember that 'good enough" but real is way better than "ideal" but not realizable.



# Conclusion

- Action Guides
  - Provide visual models or templates that help
    - Think through some aspect of architecture specification and communicate architecture decisions
    - Make the architecture team more effective
- We periodically publish action guides on our web site at
  - http://www.bredemeyer.com/papers.htm

Copyright 2001 Bredemeyer Consulting
http://www.bredemeyer.com and http://www.ewita.com

Action Guides for the Enterprise Architect
EAC San Diego, October 2001 Slide 40