Using the Open Blueprint Structure

Introduction

This document provides guidance for using the Open Blueprint structure. It describes three ways the Open Blueprint structure can be used:

1. **As a communication vehicle and checklist**
2. **As a planning tool for incorporating new functions and improvements to present systems**
3. **As a tool to evaluate or redesign an entire software architecture**

This section contains four usage examples, a summary of business situations in which the Open Blueprint structure helps, and an introduction to software architectures. The following sections give guidance on using the Open Blueprint structure.

Open Blueprint Usage Examples

Open Blueprint usage briefs provide examples of how businesses have benefited from using the Open Blueprint structure. Four short summaries are provided below.

Oregon's Department of Transportation's architecture was in need of redesign to better accommodate a new drivers licensing application with advanced technologies. Technology consultant Claudia Light implemented the new overall system architecture and says: "I honestly feel that if we hadn't taken the Open Blueprint approach, we would still be working on the original draft we published over six months ago."

Citizens Utilities had grown through acquisitions, and it was difficult for the IT organization to support the business as a whole. The business developed "Vision 2002", a plan for operating the company, and Nick Ioli, CIO, developed the supporting IT integration plan. "The Open Blueprint presents us with a road map that touches on all of the technical issues we needed to address."

Willis-Corroon lacked a consistent systems and application approach following a merger. Vincent Culhane, CEO of their Information Services Organization, based their technical architecture on the Open Blueprint. He estimates that savings in the implementation phase will result in $11 million over 5 years.

Appleton Papers needed an enterprise-wide resource planning system. In addition they had to analyze year 2000 problems, and they felt their systems were not flexible enough. John Tucker, VP of Information Systems, says: "The Open Blueprint structure helps us deliver applications and systems with fewer opportunities for missteps and at a minimal cost."

Business Summary

The following table shows examples of business situations in which the Open Blueprint can be used:

Business Manager's Critical Business Issue	IT Manager's Related Issue	Solution, How Open Blueprint Helps
• No view of a customer's relationships across businesses	• "Islands" of automation • Little integration and data sharing	• Provides guidelines to build integrated applications and systems
• Processes can't change quickly	• Systems and applications are difficult to change • New technologies are difficult to integrate	• Promotes quick modular changes through a building block approach • Documents how to integrate different technologies into today's base
• Can't assimilate mergers and acquisitions quickly	• Can't easily reconcile systems and application needs	• Provides a plan to identify and reconcile change • Helps select a common technological base

What is a Software Architecture?

An architecture usually includes definitions for standardized building blocks and rules for combining them. A software architecture is an organized set of software building blocks. For each building block, two properties are important:

- What it does (functions)
- How it relates to other building blocks (interfaces)

The benefits of using standardized building blocks defined by a software architecture include:

- Improved user productivity resulting from a consistent user interface, integrated applications, and data sharing
- Improved development efficiency resulting from the use of common components and products
- Reduced lifecycle costs, such as reduced duplication, support for incremental growth and replacement, and reduced training costs
- Improved security
- Improved interoperability
- Improved portability and scalability
- Increased vendor independence
- Improved manageability, including reduced operation, administration and maintenance costs

These benefits can result in shorter time to market and the ability to support more users and customers without increasing resources.

Concepts and Terminology in the Open Blueprint

The Open Blueprint structure is illustrated in Figure 1. It defines the components that are the basic building blocks of the software architecture (for example Relational Database and File). It defines their functions and shows how they relate to other components. It groups them into logical entities (for example Data Access Services). Each component provides specific services and can use other components' services. Components connect using APIs and protocols. A component's function can be implemented using several different technologies. The Open Blueprint structure incorporates selected technologies for the components.

Applications and development tools use Open Blueprint services. In practice, listing your applications in the Open Blueprint application area helps visualize how the Open Blueprint relates to your situation.

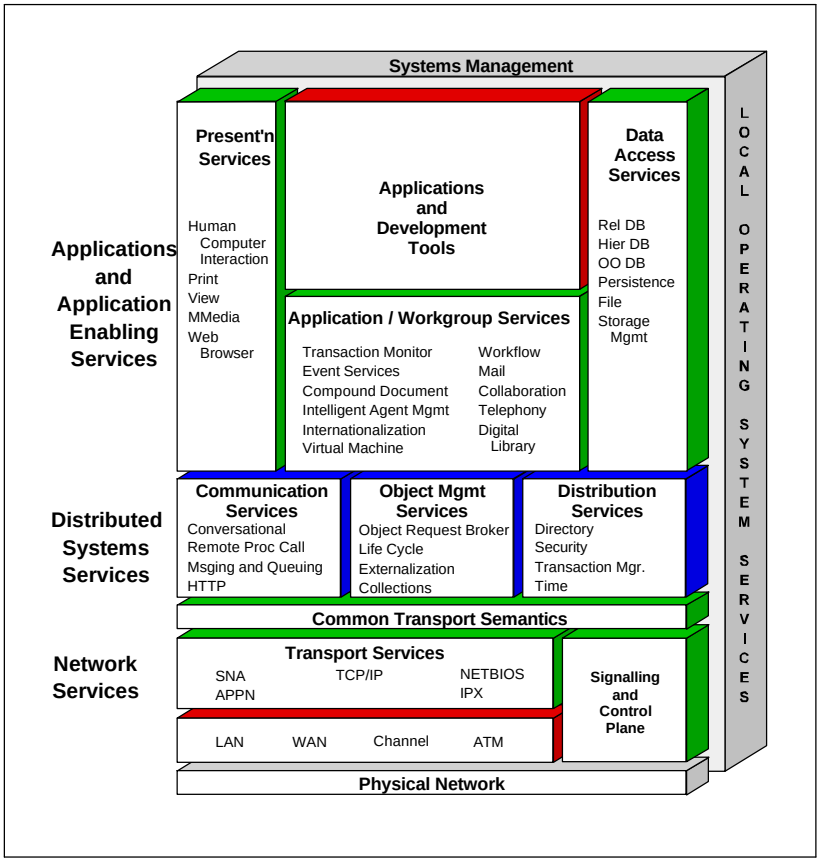

Figure 1. The Open Blueprint Structure

Sub-Assemblies

In practice, a software architecture is typically implemented by defining sub-assemblies. Sub-assemblies are constructed using a combination of components. For example, a data server sub-assembly can include relational database, file, security, conversational, and selected network services components. Other examples of sub-assemblies include application servers, Web servers, firewalls, and clients. Figure 2 describes which Open Blueprint components are chosen for these sub-assemblies. In addition to technical function, sub-assemblies can be defined by application function such as billing sub-assembly.

The attached worksheet helps document the content of your sub-assemblies

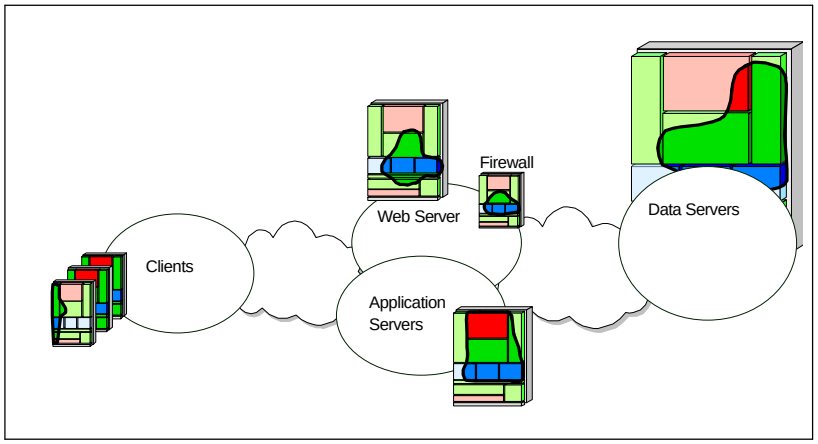

Figure 2. Examples of sub-assemblies that consist of a sub-set of Open Blueprint components

Using the Open Blueprint as a Communication Vehicle and Checklist

When you start planning additions and changes to existing applications and systems, you usually start with meetings where ideas are presented and evaluated and finally a development direction is determined. The Open Blueprint provides a vocabulary and context that help make these meetings and other planning activities more effective.

Common Vocabulary and Context

Diverse business units and organizations require clear and consistent communication.
Using the Open Blueprint structure helps communication by providing:

- A picture that makes the software architecture visible and tangible
- A common vocabulary
- A context in which to discuss and organize development issues
- A structure that enables people to work at different levels of detail

Checklist

Distributed applications and systems require multiple components that need to be integrated and to work together. The Open Blueprint structure can act as a checklist that helps determine:

- If all components have been considered
- How different development tasks depend on each other
- How a development project can be divided into pieces that have defined boundaries
- If all work items have been identified

When the development direction has been set, you can plan your project in more detail. The following chapters describe how the Open Blueprint structure can support these activities.

Using the Open Blueprint Structure as a Planning Tool

This section provides guidelines and examples of how the Open Blueprint structure can support planning activities for incorporating new functions and improvements to current systems.

In general, planning activities answer four general questions, which are presented in the following table. The right hand column provides a corresponding planning activity.

1. What should be done?	Break down your improvement project into manageable pieces using the Open Blueprint structure
2. How should it be done?	Identify the required software functions, determine what to build and buy, and evaluate proposed software.
3. How much work is needed?	Estimate the effort required to implement the improvements
4. When can it be ready?	Prepare a time line for the improvements

The following sections describe detailed work items for each planning activity:

1. Break Down Improvement Project into Manageable Pieces

To break down your project into specific, manageable pieces, select all components that are affected by your improvement project from the Open Blueprint structure. Include your applications as components in Open Blueprint's Application area. Then, classify your selected components and relationships as illustrated in Figure 3 below.

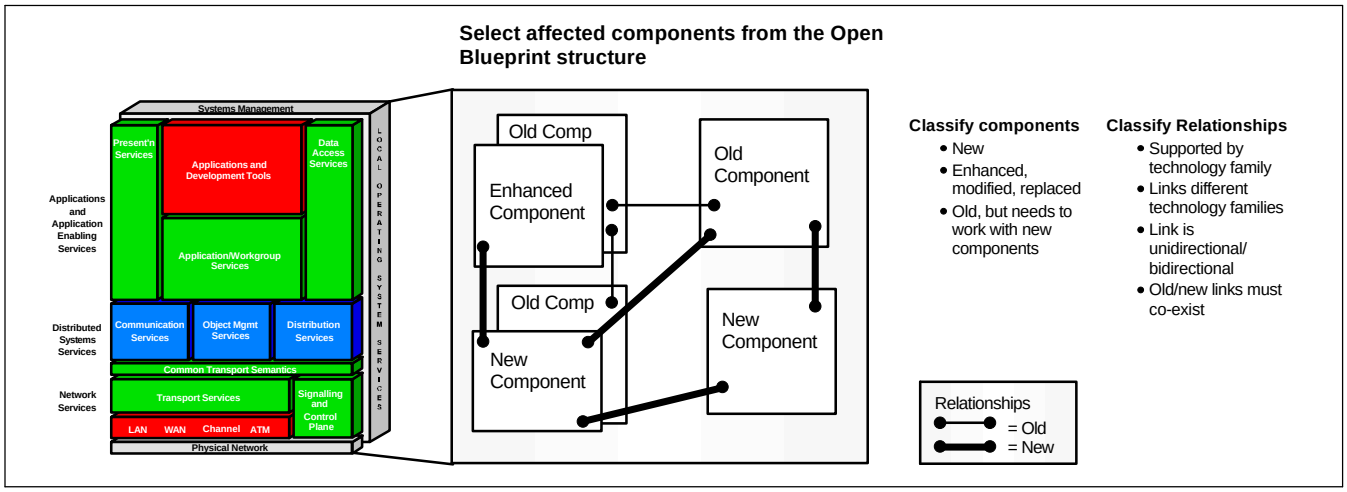

Figure 3. Structuring an improvement project

Example

Suppose you want to make improvements to your order entry applications so that mobile salespeople can take orders working off-line and customers can order spare parts using the Web. In addition, you want these two improvements to use the same network services. To plan these improvements, you have to break down your environment to the component level, plan the common network services and define how the they are used by these two improvements.

To identify required components and relationships, answer the following questions:

Components:

- Which new components are needed (e-mail/replication process for salesman's computer, a new order entry application, a work flow process, and so on).
- Which existing components will be enhanced (Web-based user interface to order entry application, communication services, and networking).
- Which existing components need to work with new components (existing product databases must support the new technologies in the order entry applications).

Relationships:

- How do new Web forms interface with existing order entry applications? Where is the bridging and interfacing done and which technologies are involved?
- Do you need additional links to existing components (how do you access spare parts from your application; do spare parts use different database technologies than normal products)?
- Which interfaces to components will be new or modified (new security model will be added and linked to today's access control).

Having identified affected components and relationships, you have broken down your project to more manageable pieces. It is now easier to plan how each component should be enhanced , and to identify the work items and the project structure.

2. Identify Required Software Functions and Determine How to Implement Them

When you have identified the required components for your improvement project, you can use the Open Blueprint documentation as a reference to define the functional requirements for software products that implement them. You can also use the Open Blueprint documentation to define dependencies and interfaces to other components. Then, plan the software structure and determine what to build and buy. Make sure the proposed software:

- Contains required function
- Supports selected APIs and protocols
- Supports related components (such as systems management and security)
- Does not impose unnecessary duplication of present functions

3. Estimate the Effort Required

After completing the first two steps, estimate the work effort:

- Define all changes in components and relationships as work items
- Prioritize and sequence work items
- Estimate workload and impact on people
- For large projects, walk through the steps in "Developing a Tailored Software Architecture"

4. Prepare a Time Line

Having defined your work items and estimated the effort required to complete them, prepare a time line:

- Create a logical dependency diagram using the relationship information
- Map this diagram onto a project plan
- Estimate the duration and timing of your work items based on your work estimates, available resources, and possible system update times

Examples of Improvement Projects where Open Blueprint Helps

Figure 4 illustrates five typical improvement projects in which the Open Blueprint structure helps:

- Adding a centralized customer database
- Adding Internet access
- Adding object-based applications
- Making EDI an independent application component
- Making applications more modular and component based

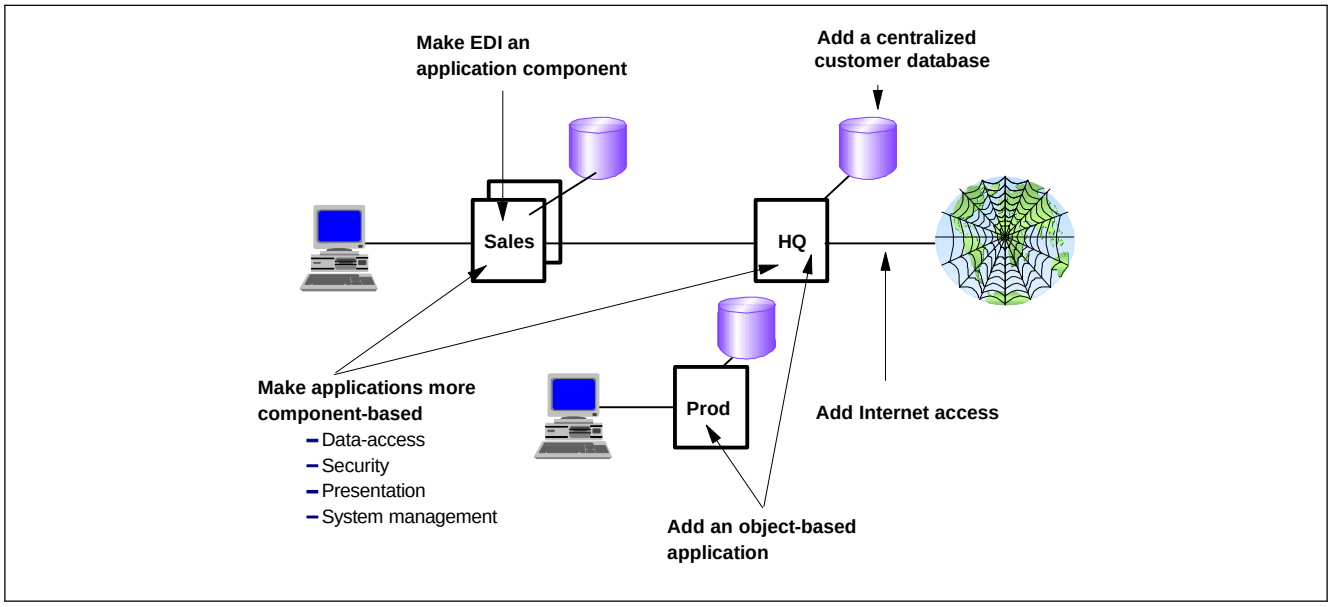

Figure 4. Examples of improvement projects in which the Open Blueprint structure help

Add a Centralized Customer Database to the Current System

To create a centralized customer database that will be used by 15 systems in 6 geographical areas, and to make the database accessible by marketing applications and accounts receivable applications, you need to use the previously described planning methodology and:

- Make sure that the database technologies are supported by all participating systems
- Make sure related components such as directory, security, and data management support the required data access
- Make sure that the database can be accessed with satisfactory response times from all 15 systems

Add Internet Access to Your Applications

To enable access to your current data and applications through the Internet, you need to:

- Define the functions required for Internet access (browse, query, update, transaction processing, security)
- Define all dependencies and relationships
- Compare your requirements to the Open Blueprint documentation for function, security, and links to other components such as transaction monitor and systems management
- Build a function/technology/interface diagram that documents your building blocks
- Evaluate proposed Internet access solutions against these requirements
- Consult the Network Computing Framework for suggestions on building Internet applications

Add an Object-Based Application to Your Current System

To add object based applications, use the Open Blueprint structure as your software services map and:

- Define which services and components the application will use or interface with
- Define which services are used as integrated object services and which services are accessed using object wrapper functions and gateways
- Define the object service to be used. Compare your plans with the Open Blueprint on the use of Object Request Broker, Object Services, links to security and so on
- Check the object interfaces to the components that provide object interfaces, for example the components that provide presentation services and data access
- Define object interfaces to components that include wrappers or gateways

Make EDI an Application Services Component

To add a technology to your architecture such as EDI (Electronic Data Interchange), you need to:

- Define the functions of this component (building block)
- Describe how this component relates to other components in your architecture (file, security, network and so on)
- Define the API interface that describes how this building block is used

Make Applications More Modular and Component-Based

Separating application services from applications makes applications smaller, easier to maintain and more portable. The following services can easily be provided by external modules: data-access, presentation, security, and systems management services. To externalize application services from your applications:

- Define the services that will be provided by external components
- Using the Open Blueprint documentation as a reference, carefully specify the functions, technologies and interfaces that these external components provide
- Update your application design to make use of the externalized functions and interfaces

Developing a Tailored Software Architecture Using the Open Blueprint Structure

The Open Blueprint is a general purpose, open architecture for heterogeneous distributed environments. It can be tailored to a specific purpose and environment. An architecture project is often tied to a business transformation project. Clear business goals and management support are required for a software architecture project to be successful.

Your definition of an architecture should include:

- A description of “target state”
- Building blocks
- Principles and rules
- Criteria for selecting architecture options, products, and so on
- A management process.

The process for developing a tailored architecture is outlined in Figure 5.

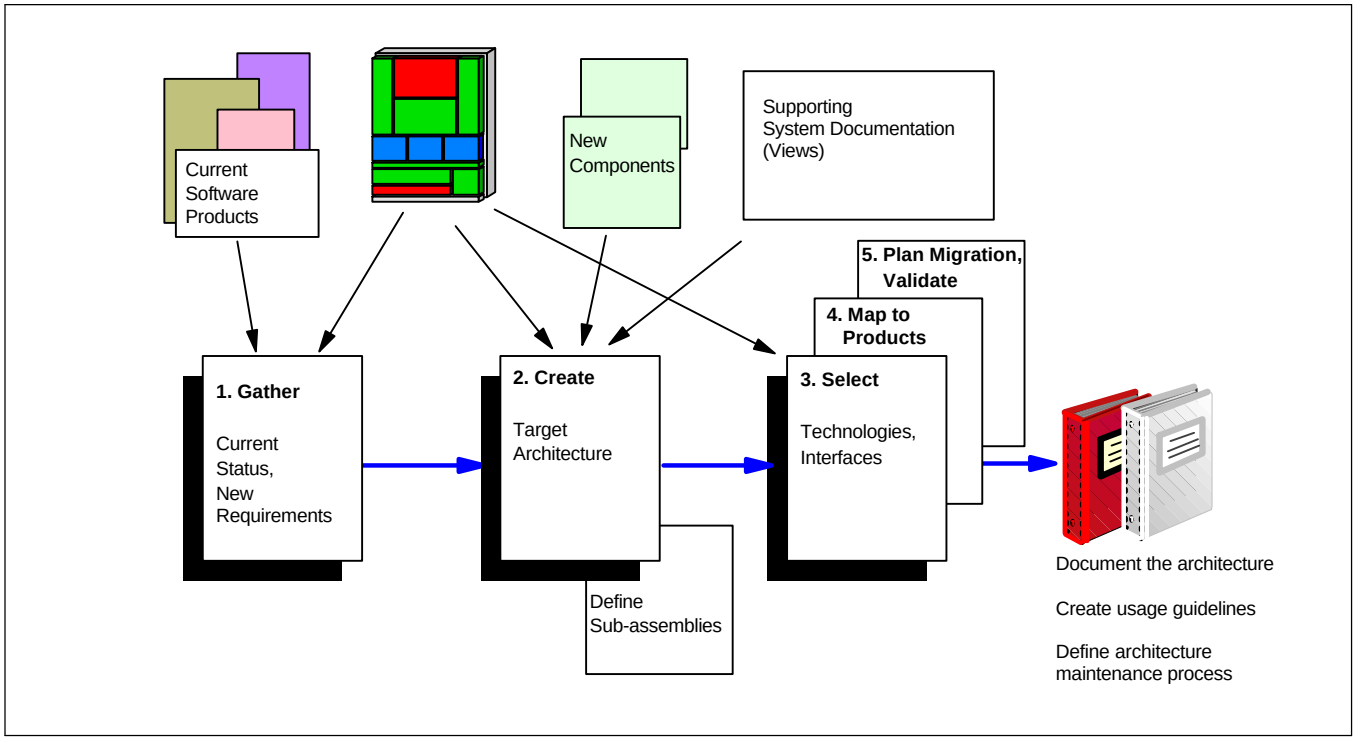

Figure 5. Architecture development process

1. Gather the Current Status and Requirements

The Open Blueprint structure provides an excellent framework for gathering data and documenting your current environment in different levels of detail. This phase identifies the problems, requirements, constraints and the technical base to keep. This phase contains the following steps:

A. Document the current IT environment by mapping it to the Open Blueprint structure:

- Select the Open Blueprint functional services and components you have or need
- List the software products that provide these services

- Analyze these products and their use:
  - What is a product's function (compare to Open Blueprint components)
  - How does it relate to other components (compare to Open Blueprint relationships)
  - What technologies does the product provide
  - What APIs and protocols does the product provide or use
- Document your environment using the Open Blueprint service and component names as your table of contents

B. Analyze the current IT environment and summarize findings. Determine:
- Problems
- Constraints
- The IT base you want to keep

C. Determine key requirements and define them as building blocks.

2. Create a Target Architecture

During this phase you define component functions, relationships, and sub-assemblies and verify their usability. Before defining the new architecture, you need to determine and document the external factors that influence your architecture design:
- Create an IT vision that describes how IT supports your business
- Document your guiding principles such as the centralization and decentralization principles that affect your design
- Establish the rationale for making choices and develop evaluation criteria

Create the new architecture by defining its components, structuring them, and defining sub-assemblies:
- Select the Open Blueprint services and components that you need
- Document the functional requirements of the components. You can use the Open Blueprint as a reference to document your selection and differences
- If needed, define new components and document them following the Open Blueprint model
- Document how components depend or relate to other components in your architecture
- Define the structure of your architecture
- Define your sub-assemblies in similar way that you defined your components

Check the architecture using architectural views. An architecture and its development process are usually supported by different views of the current and future systems. The views are used to verify that the architecture is complete and applicable to the future environment. The views include functional views and technical views that support architecture implementation. The most common architectural views include:
- A user view that shows what applications do (functions)
- An application view that documents the scope, integration requirements, security, volumes, and so on
- A data view that documents the flow of data as it is stored and processed
- A systems view that documents the systems, the distribution of function geographically, and the different types of sub-assemblies such as clients, application servers, Web servers, and database servers
- A network view that documents bandwidth requirements and access and performance constraints
- A security view that assists in designing and verifying security functions
- A systems management view that documents systems management functions and relationships

3. Select Technologies and Interfaces

During this phase you select technologies and interfaces for the target architecture and verify their applicability, interoperability and integration. After selecting the architecture components, you need to select technologies, APIs, and protocols, and to consider performance aspects and product availability.

Select technologies and interfaces and evaluate selections:

- List your technology options from your current base, the technologies selected from the Open Blueprint structure, and technologies required to support your new components
- Select technologies, interfaces and protocols for the components
- Define the main interfaces for sub-assemblies
- Check that your technology and interface selections are consistent and that they enable integration using the relationship information of the components. Use system views to illustrate the impact of a specific selection.

4. Map Products to Your Architecture

During this phase you select products and define guidelines for their use. Choose the products and select the properties that implement your architecture:

- Map your architecture to the software and hardware layout that is described by sub-assembly documentation and system views
- Select products that implement sub-assemblies and provide the services
- Select products that exploit your sub-assemblies and services they provide
- Choose conforming products for each platform
- Document which product properties will be used and which will not be used

5. Plan Migration

During this phase you produce a feasibility analysis, cost and benefit documentation, risk assessment, and a migration plan. These results should be communicated both to decision makers and the technical staff that implements the migration.

Migration planning provides a reality check of your new tailored software architecture. The following list outlines the tasks required to plan migration from the current architecture to new:

- Define the migration project and classify your components and work items as:
  - New component to be developed
  - Components that must be enhanced, modified, or replaced
  - Existing components that need to work with new or modified components
  - Old and new versions of components that must coexist
- Identify the cost and benefits of each work item
- Evaluate constraints and risks
- Prioritize and sequence activities
- Redesign the target architecture if needed
- Develop phasing and migration plans

6. Define Architecture Usage Guidelines and Maintenance Process

After developing a new software architecture you should:

- Establish guidelines for using the architecture
- Define how the plan will be kept up to date
- Define and document the plan management process

Summary

This document has described some of the most common ways the Open Blueprint can be used. The appendix lists sources for more information, including the examples of how others have used the Open Blueprint. It also describes how to get started and includes a worksheets that can help you use the Open Blueprint.

The IBM Consulting organization has performed a number of assignments where Open Blueprint has been used. You may use their experience and support when planning improvements to your present systems and architectures.

Appendix

Getting Started

- Become more familiar with the Open Blueprint concepts.
  Read the Open Blueprint Executive Summary and Open Blueprint Technical Overview (GC23-3808). You can browse or download these documents at
  http://www.software.ibm.com/openblue/papers/execoview.htm
  http://www.software.ibm.com/openblue/id1b1/cover.htm
- Review Internet related concepts by reading the Network Computing Framework paper, which you can browse or download from
  http://www.software.ibm.com/ebusiness/ebusiness
- List your relevant development issues with which Open Blueprint can help
- Complete the attached worksheets for the sub-assemblies that are appropriate to your application or system.
- Use this material to develop a prototype case. Document the case and benefits of the methodology.

Examples How Others Have Used the Open Blueprint

The following usage briefs describe how businesses are using the Open Blueprint structure:
(also available on the web at http://www.software.ibm.com/openblue/howuse.htm)

1. Oregon Transportation Department prepares for the future, G325-5103
2. Willis Corroon Group develops Information Service plan for the future, G325-5105
3. Citizen Utilities: Dynamic company embarks on bold journey, G325-5107
4. Open Blueprint stacks up for Appleton Papers, G325-5109

Worksheets

Figure 6. Open Blueprint - Major Interfaces and Protocols - illustrates the technologies and interfaces contained in the Open Blueprint. It is useful in helping you with the selection of technologies and interfaces for your environment.

The worksheet lists Open Blueprint components and provides space for data collection. It is useful in the initial phases of gathering current status and defining sub-assemblies.

You can start by filling out one worksheet for each of your major applications such as order entry. The order entry applications might use relational database, mail, file, transaction monitor and so on. Write down the technologies used for these components. Once you have completed a few applications, you can create your own master worksheet which includes the key technologies used in your installation. You can also use the worksheet to document components and technologies that form your sub-assemblies.

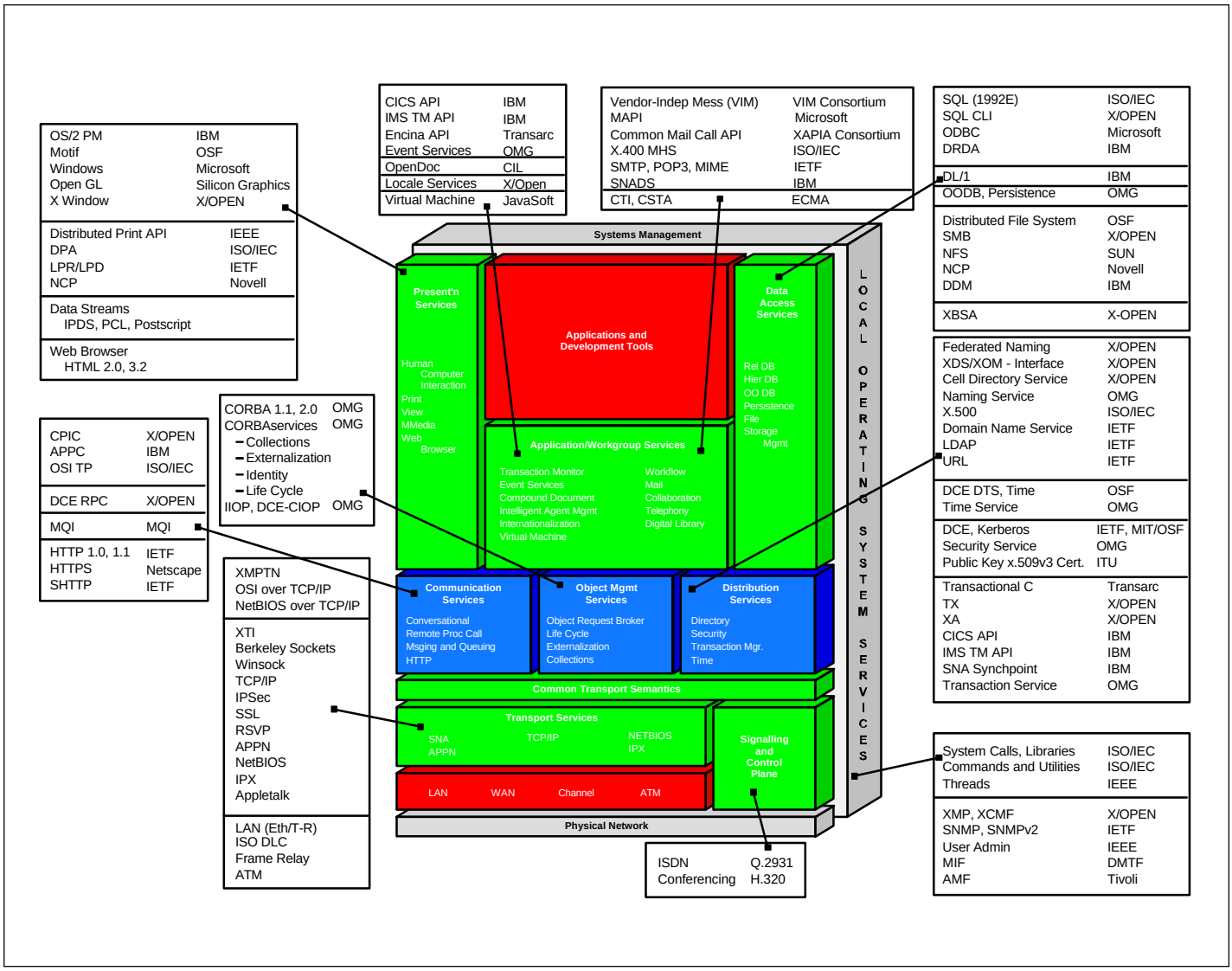

Figure 6. Open Blueprint - major interfaces and protocols

Open Blueprint Worksheet #3

Applications and Applic Enabling Services

Presentation Services

Human/Computer Ia.
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

Distributed Print
- ☐ _____
- ☐ _____
- ☐ _____

Print Data Streams
- ☐ _____
- ☐ _____
- ☐ _____

View
- ☐ _____

Multimedia
- ☐ _____

Web Browser
- ☐ _____

Distributed Application

_____

_____

_____

_____

Development Tools

_____

_____

Application Services

Transaction Monitor
- ☐ _____
- ☐ _____
- ☐ _____

☐ **Event Services**
☐ **Compound Document**
☐ **Intelligent Agent**
☐ **Internationalization**

Virtual Machine, Java
- ☐ _____

☐ **Workflow**

Mail
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

☐ **Collaboration**
☐ **Telephony**
☐ **Digital Library**

Data Access Services

Relational Database
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

Hierarchical Database
- ☐ _____

Object Oriented DB
- ☐ _____

Persistence Services
- ☐ _____

Distributed File
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

Storage Management
- ☐ _____
- ☐ _____
- ☐ _____

Distributed Systems Services

Communication Services

Conversational Comm
- ☐ _____
- ☐ _____
- ☐ _____

Remote Proc Call
- ☐ _____

Messaging and Queuing
- ☐ _____

HTTP
- ☐ _____

Object Management Services

Object Request Broker
- ☐ _____
- ☐ _____

☐ **Object Services**

_____

_____

_____

_____

- ☐ _____
- ☐ _____

Distribution Services

Transaction Mgr.
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

Time
- ☐ _____
- ☐ _____

Security
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

Other Security
- ☐ _____
- ☐ _____
- ☐ _____

Directory and Naming
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

Other Directory
- ☐ _____
- ☐ _____
- ☐ _____

Common Transport Semantics

☐ _____ ☐ _____ ☐ _____ ☐ _____

Network Services

Transport Network

☐ SNA ☐ TCP/IP ☐ Net BIOS
☐ APPN ☐ _____ ☐ IPX
☐ _____ ☐ _____ ☐ _____

Signalling and Control Plane

- ☐ _____
- ☐ _____

Subnetworking

☐ LAN (Eth/T-R) ☐ _____ ☐ _____ ☐ ____

Physical Network

Systems Management
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____
- ☐ _____

Local Operating System Services
System calls, Libraries
Commands and Utilities
Threads

For more info: handbooks G326-0359, GC23-3808, SBOF-8702/SK2T-2478
http://www.software.ibm.com/openblue

S12116P01.01 FL