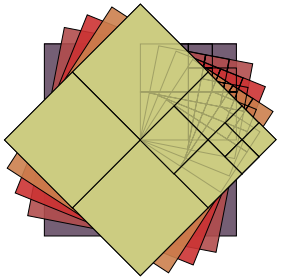


Banyan

Information Management System

The power of collaborative, clear thinking

Developing a Web Based,
Collaborative,
Semantic Information Manager

R. H. Barter

December 14, 2005

This work was performed under the auspices of the U.S. Department of Energy by the University of California Lawrence Livermore National Laboratory under Contract No. W-7405-Eng-48.

Lawrence Livermore National Laboratory, P.O. Box 808, Livermore, CA 94551

Overview

- Banyan is a Product and a Process for solving complex problems
- Manages engineering information for two weapon systems
- Directly applicable to assurance management

Banyan
Information Management System

Banyan Manages Documents, Information, & Semantics

Document


DOE/NNSA
WEAPON QUALITY POLICY (QC-1)

Information

2.1 Risk-BAsed Program

requirements. An organization shall use a risk management process when choosing to apply the "shall" requirements of this document in a graded (tailored) manner and document that process in their QAP or WQAP.

Semantic Context

Information

4. OBJECTIVES

- Apply QA measures to work activities in a cost-effective manner based on the maturity, applicability and risk associated with the activity

Document


Defense and Nuclear Technologies

Directorate

QUALITY ASSURANCE PLAN

February 2002

Lawrence Livermore National Laboratory

Policy

Background

- Search for an engineering information management tool
  - Commercial (closed): general purpose, application specific
  - Open Source
  - Standalone
  - Client-server
  - Web based
  - Technologies
    - Java
    - Perl, PHP
    - Python
- Narrowed the search (technology watch)
  - Java/struts/tomcat
  - Python/Zope/Plone
- Along came ADAPT

  NNSA, Advanced Design and Production Technologies

Background

$3M ADAPT project – developed for a Life Extension Program

- 12,000 documents
- 1,300 information elements
- 700 risk-based activities

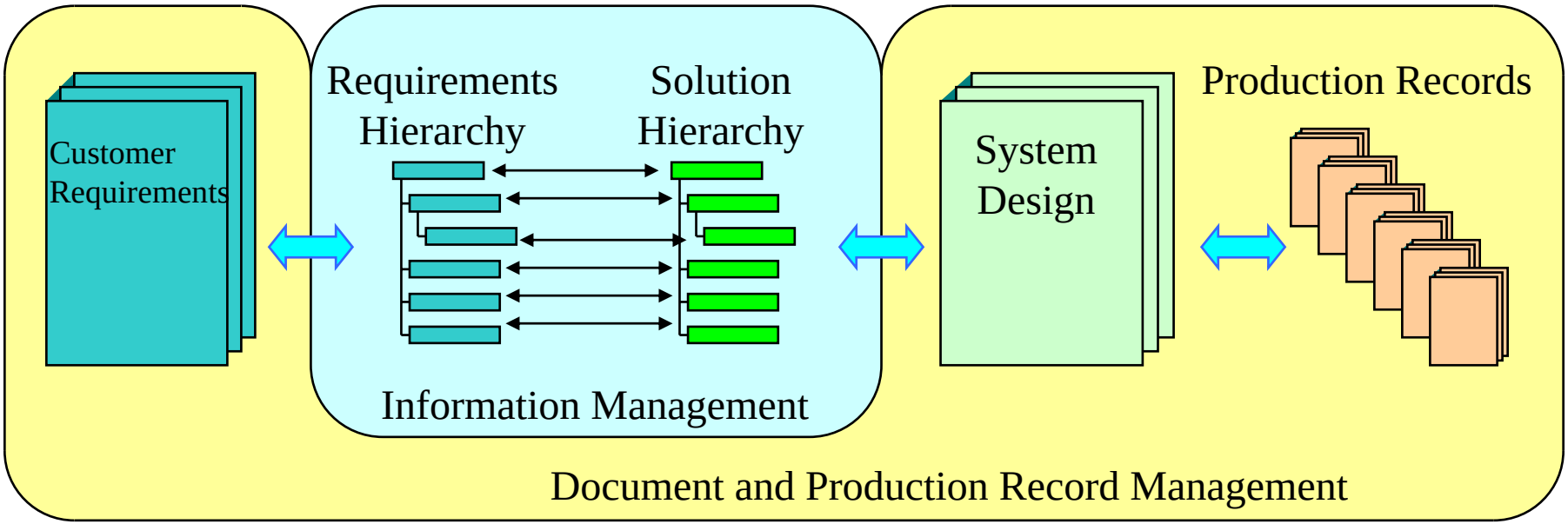
Banyan paid for itself in its first month of use through the risk-based prioritization of certification activities

Our Experience in W-Program

- Required for certification
- Unambiguous traceability
  - Customer requirements
  - Tests
  - Simulations
  - Surveillance activities
- We are playing a key role in RRW

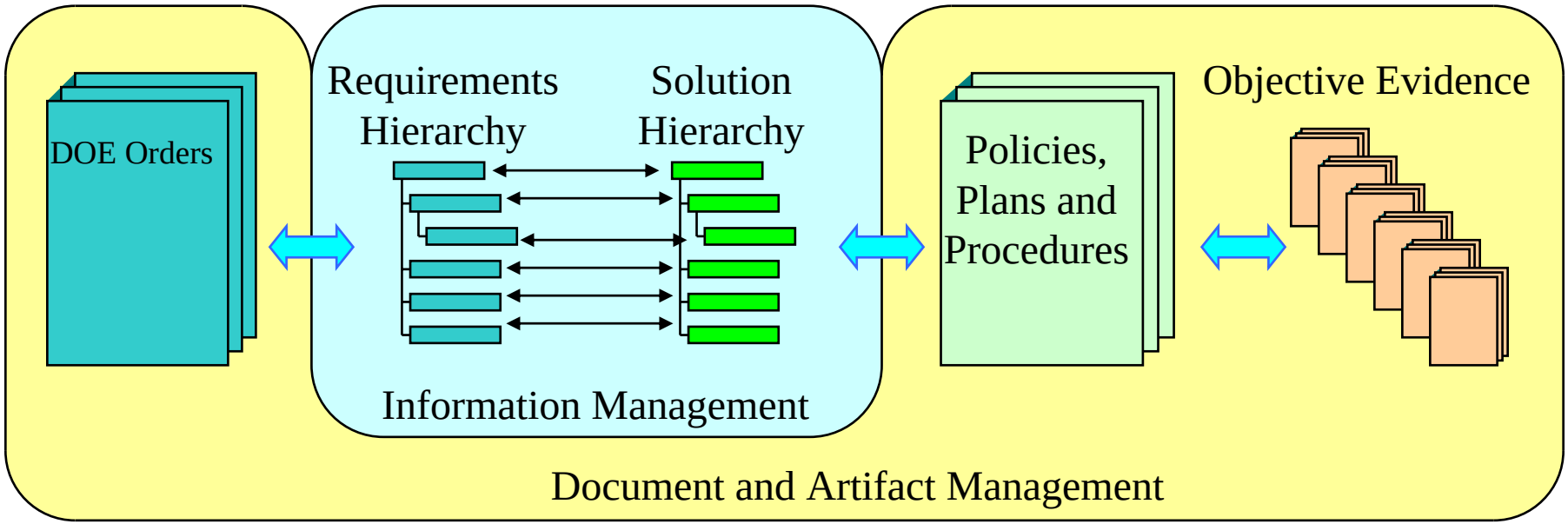
Unambiguous traceability forms the backbone of the certification effort

Banyan as an Engineering Tool



Unambiguous traceability

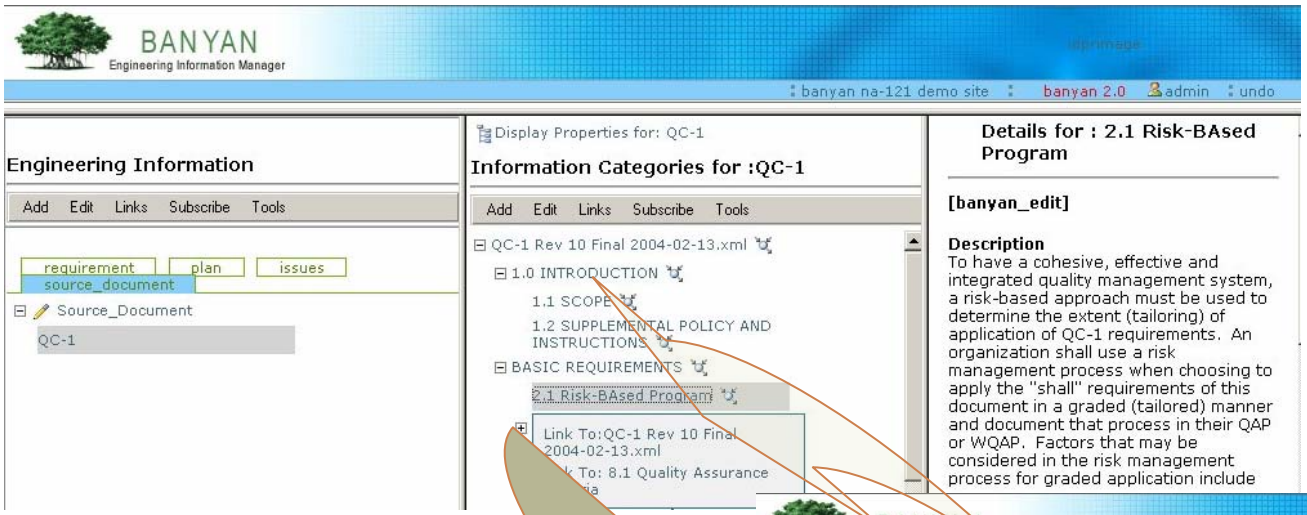
Banyan as an Assurance Manager

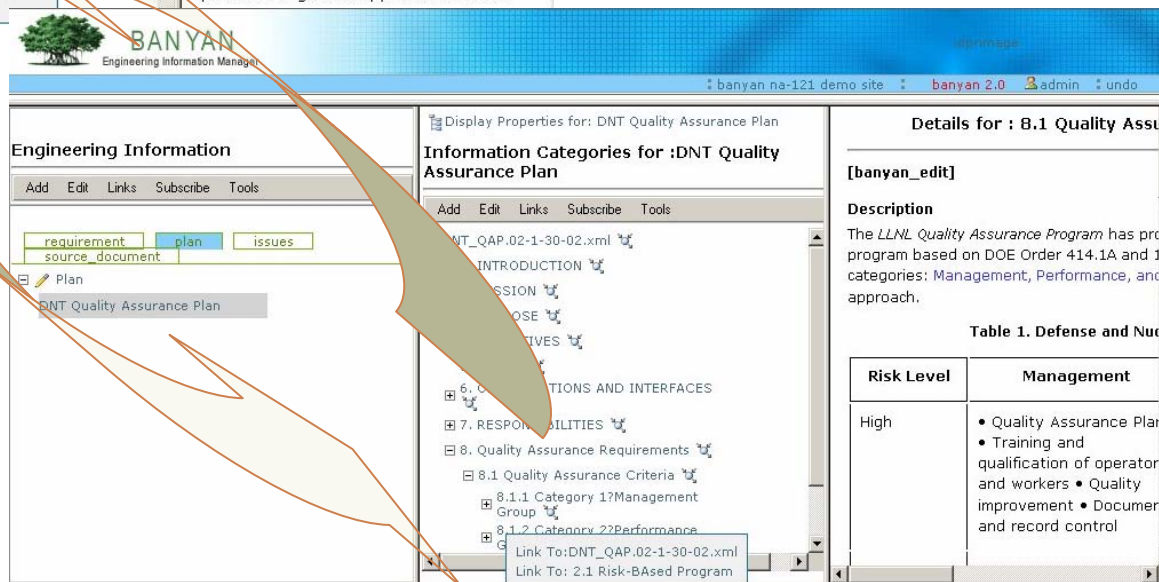


Audit-proof Compliance

Banyan
Information Management System

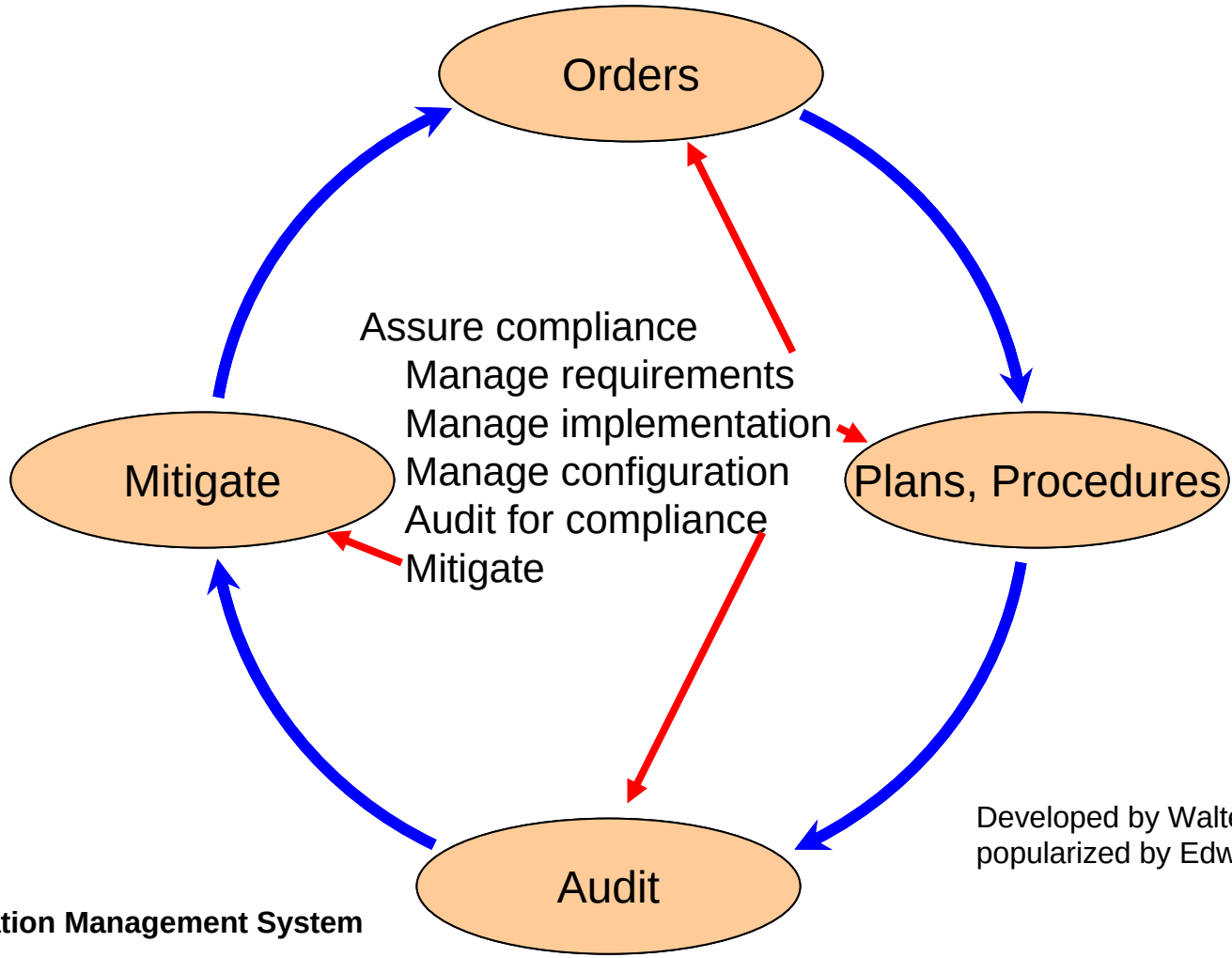
Unambiguous Traceability





Banyan Information Management System

Banyan supports management processes

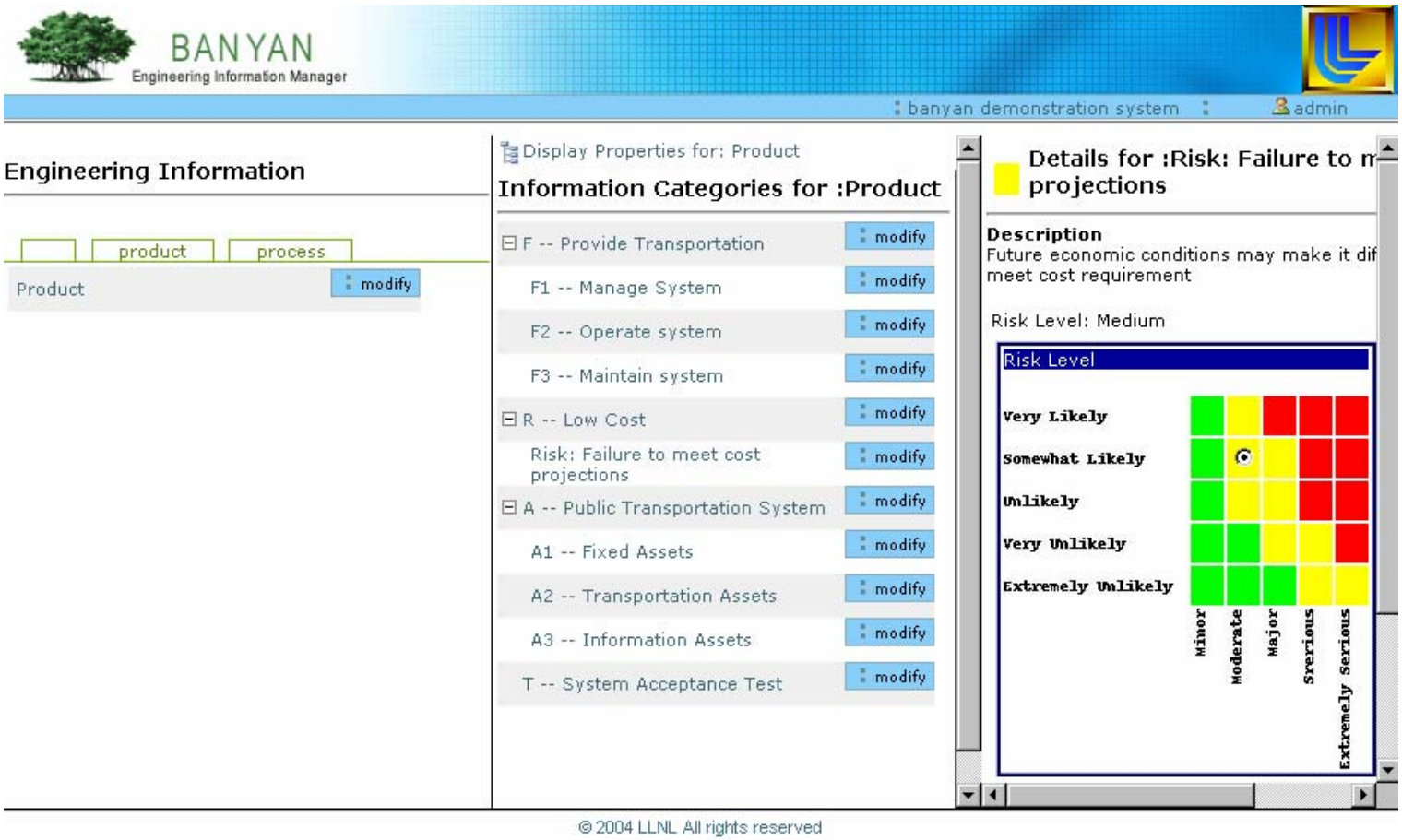


Developed by Walter Shewhart and popularized by Edwards Deming

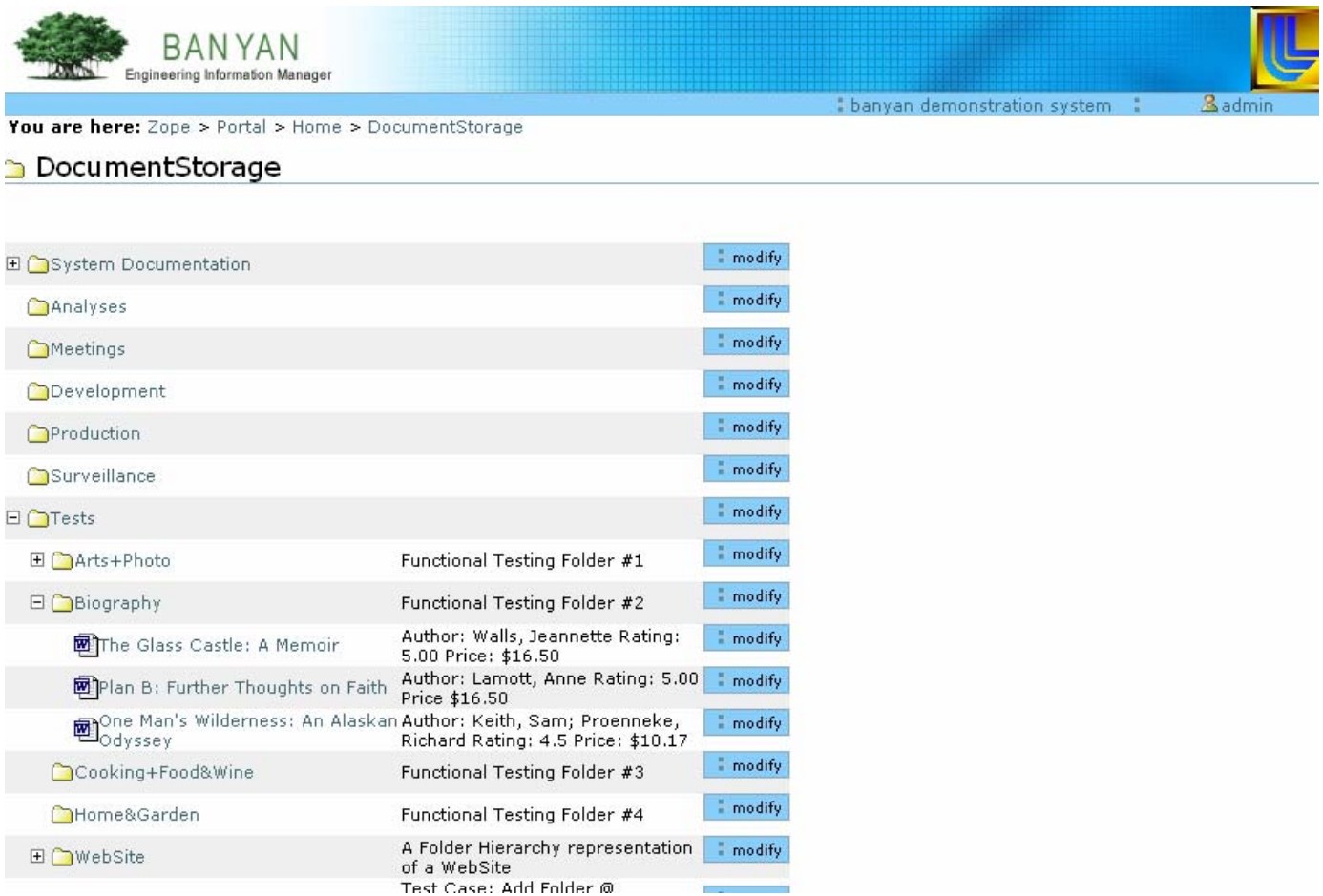
Front Page


BANYAN
Engineering Information Manager
: banyan demonstration system :
admin
Banyan Demonstration System
: Engineering Section
: Document Storage Area
System Documentation
Analyses
Meetings
Development
Production
Surveillance
Tests
: Engineering Section Reports
: Search Documents for:
search
: Search Document Storage Area
: Search Engineering Section
: Configuration
© 2004 LLNL All rights reserved

Engineering Section



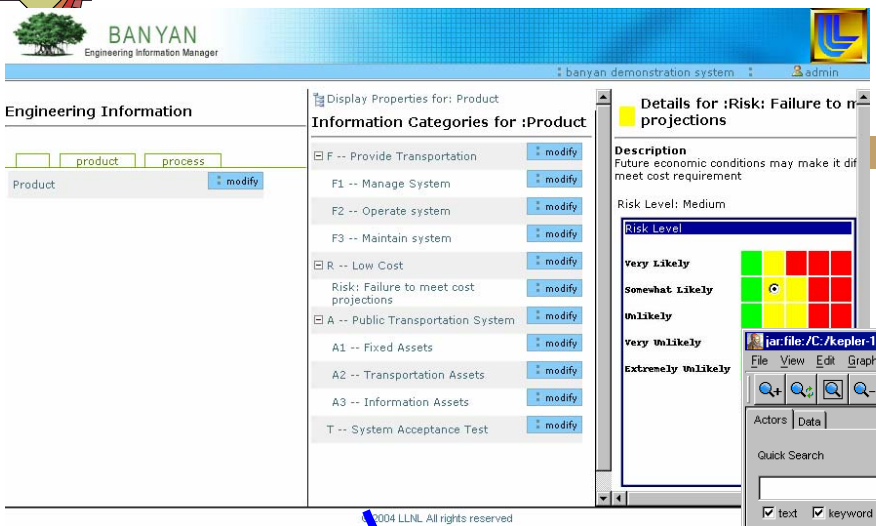
Document Storage



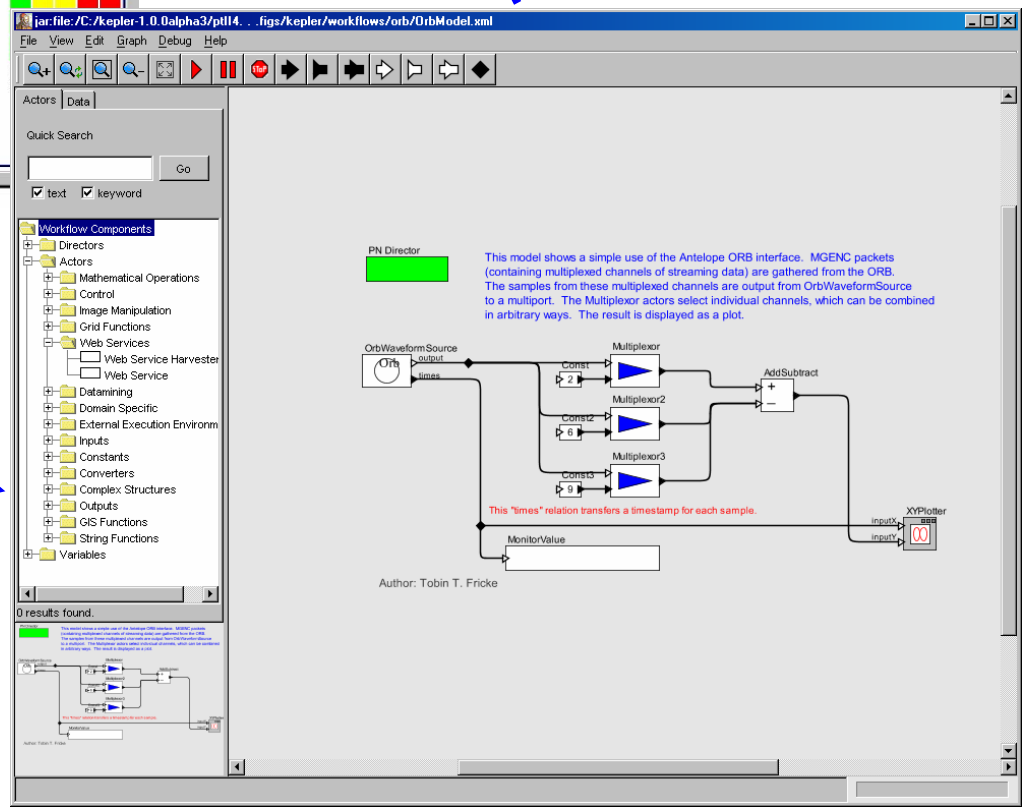
Process Modeling with Kepler

Banyan
Information Management System

Web Services



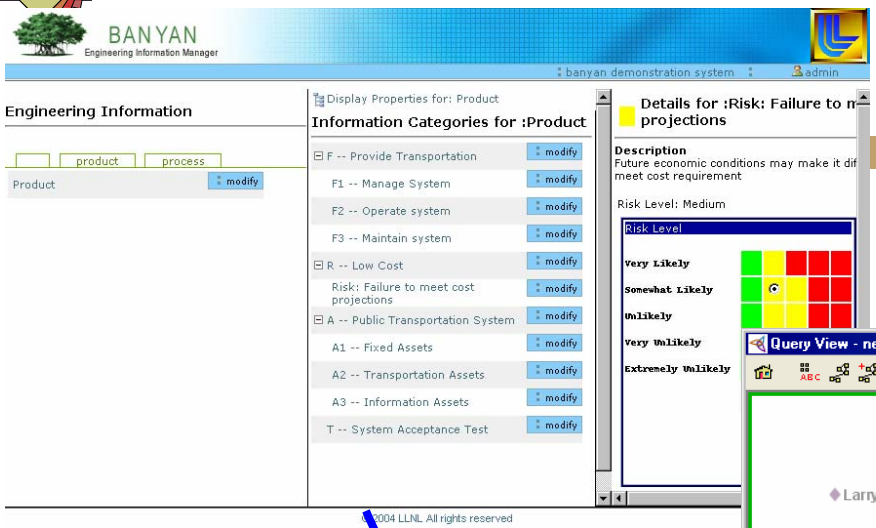
Web Services



Web Services

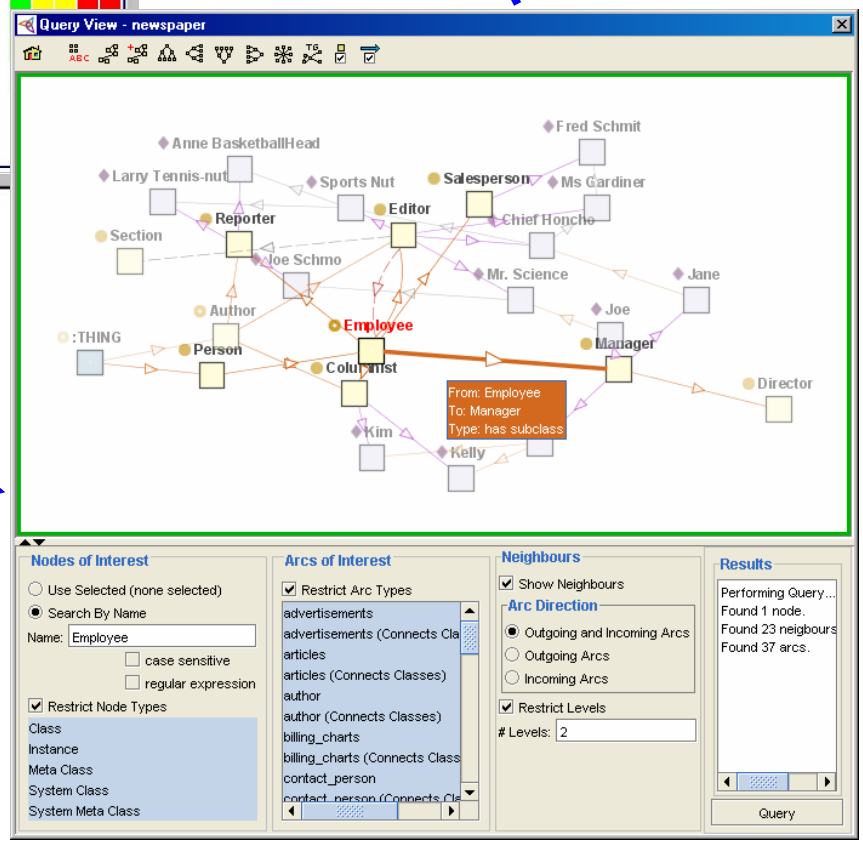
Ontology Manipulation with Protégé

Banyan
Information Management System

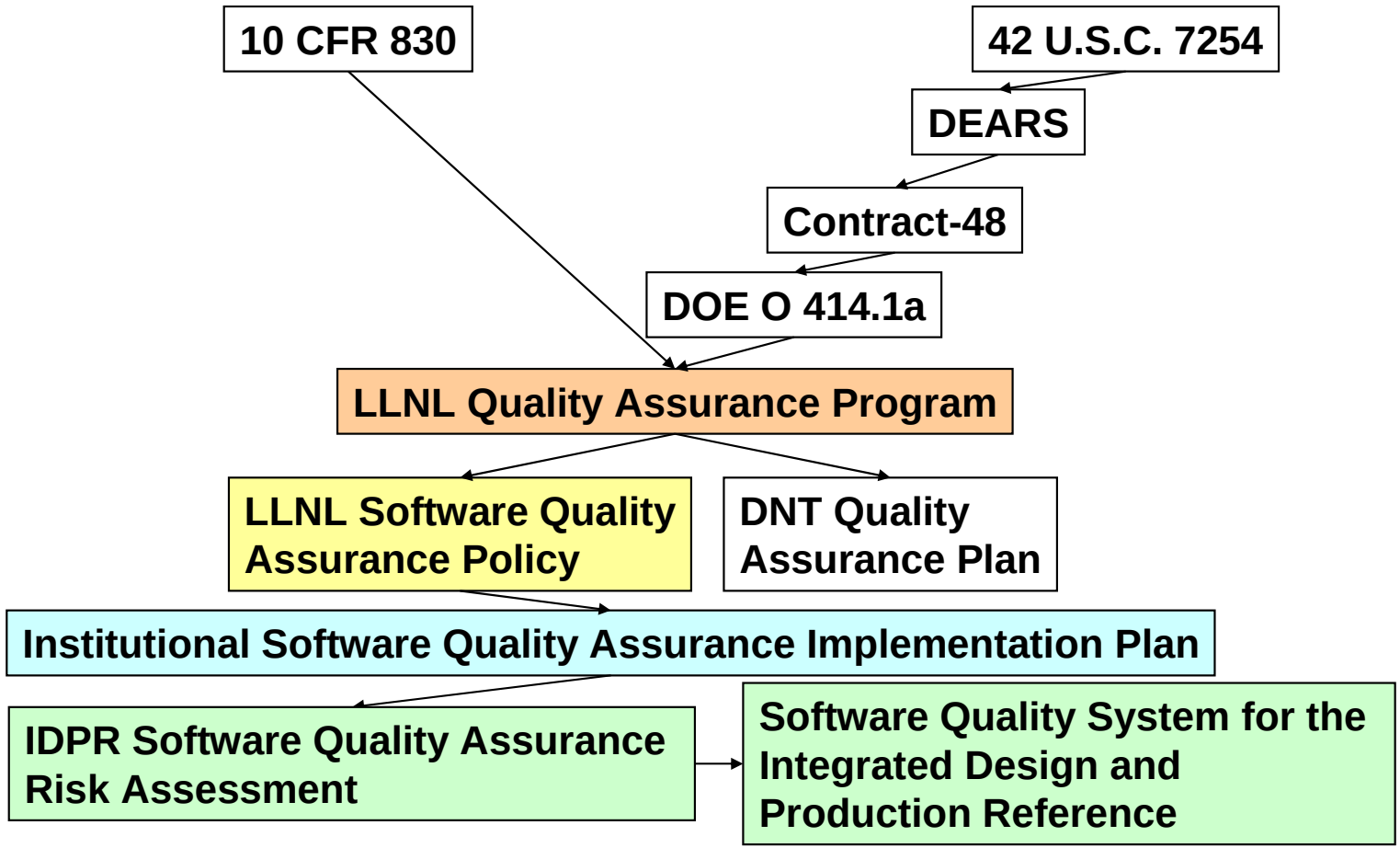


Web Services

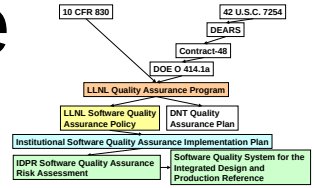
Web Services



A quality relationship



Consequence of failure is Tier 3/4

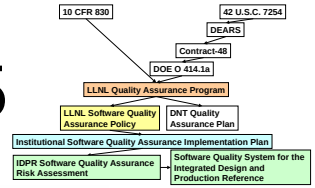


Annotated Risk Consequence Tiers from the Institutional Software Quality Assurance Implementation Plan

Consequence Tier	Risk Categories/Consequences			
	Environment, Health & Safety	Program, Cost & Schedule	Political & Public	Security
Tier 1	Catastrophe (multiple fatalities and serious injuries) Impact requiring immediate evacuation or other drastic action to protect public Felony or serious civil liability Significant hazard to public & employees Primary safety barrier	Major dollar loss, e.g. over $1M Major schedule impact Failure to meet program requirements	Major loss of public confidence International adverse publicity	Release of classified or proprietary information Damage to or theft of physical or intellectual property
Tier 2	Serious OSHA/EPA violation or civil liability Hazard to employees Low hazard & primary safety barrier & simple software or high hazard for LLNL workers & secondary safety barrier & complex software Impact requiring action by offsite public	Minor cost impact, e.g. $200K to $1M Minor schedule impact Failure to meet program goals Potential LLNL operations work stoppage	Loss of public confidence National adverse publicity	Failure to account for classified or proprietary information Failure to account for physical or intellectual property
Tier 3	Minor OSHA/EPA violation or civil liability Accident possible, fatalities or serious injury unlikely Low hazard for LLNL workers Secondary safety barrier Impact requiring offsite cleanup, but no immediate effect on public	Acceptable cost impact, e.g. $20K to $200K Acceptable schedule impact Failure to meet *minor* program goals Potential project work stoppage or LLNL operations inconvenience	Minor public concern Adverse publicity within user community	Minor security concern
Tier 4	OSHA/EPA violation, civil liability unlikely Software does not control or mitigate a hazardous operation No offsite impact Negligible impact to LLNL Employees	Negligible cost impact, e.g. less than $20K Negligible schedule impact Negligible program impact Potential for no more than a project inconvenience	Negligible public concern Adverse publicity within the Lab	No need for additional security beyond the norm

Table 4. Risk Consequence Tiers

Likelihood score is 3.875

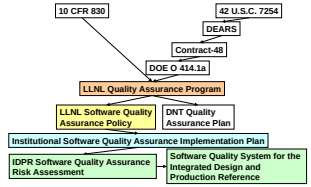


Annotated Likelihood table from the Institutional Software Quality Assurance Implementation Plan

Factors contributing to likelihood of software failure	Un-weighted likelihood of failure score					Weighting factor	Likelihood of failure[1]
	1	2	4	8	16		
Software Team Complexity	Up to 5 co-located people	Up to 10 co-located people	Up to 20 co-located people or 10 people with external support	Up to 50 co-located people or 20 people with external support	>50 co-located people or 20 people with external support	x2 **1**	**2**
Team Experience	12 or more years	6 years	3 years	1 year	4 months or less	**1** x2	**2**
Contractor Support	None	Contractor with minor tasks		Contractor with major tasks	Contractor with major tasks critica to project success	**8** x2	**16**
Organization Complexity[2]	One location	Two locations but same reporting chair	Multiple locations but same reporting chain	Multiple provides with Prim/sub relationship	Multiple providers with associated relationship	x1 **1**	**1**
Schedule Pressure[3]	No deadline		Deadline is negotiable		Non-negotiable deadline	**8** x2	**16**
Process Maturity	CMM Level 4, 5 – managed, optimizing	CMM Level 3 – defined processes	CMM Level 2 – repeatable processes	Record of repeated success	Little or no history	**8** x2	**16**
Degree of Innovation	Proven and accepted		Proven but new to the development organization		Cutting edge	x1 **1**	**1**
Level of Integration	Simple – Standalone				Extensive Integration Required	**2** x2	**4**
Requirement Maturity	Well defined objectives – No unknowns	Well defined – Few unknowns		Preliminary objectives	Changing, ambiguous, or untestable objectives	x2 **1**	**2**
Software Lines of Code[4]	Less than 50K		Over 500K		Over 1000K	**1** x2	**2**
Subtotal							**62**
							/16
Total							**3.875**

Table 5. Likelihood of Failures Due to Software Environment

The Risk Level is Low



Annotated Risk Level from the Institutional Software Quality Assurance Implementation Plan

Consequence of Failure Tiers				
Tier 1	Medium	High	High	High
Tier 2	Low	Medium	Medium	High
Tier 3	Negligible	**Low**	Low	Medium
Tier 4	Negligible	Negligible	Negligible	Low
		2	4	8	16

Likelihood of Software Failure

Table 2. Risk Level Assessment Grading

Practice 1 of 3

Practice	IDPR
Understand requirements to the extent known	The pilot and prototype systems were designed to elicit requirements.
Understand customer interactions	The pilot and prototype systems were designed to elicit an understanding of customer interactions.
Perform usability analysis as needed	The pilot and prototype systems were designed to analyze usability from the customer's point of view.
Determine applicable regulatory requirements	Possible Price-Anderson impact in the area of records retention. Implications are being investigated by A. Vinzant.
Understand software interfaces	Software interfaces have been identified.
Utilize coding standards	Specified in the software requirements document.
Select, train, and task team members	Zope training for all team members.
Identify, acquire, and deploy required resources	All open source software. Servers have been procured and workstations are procured as necessary.

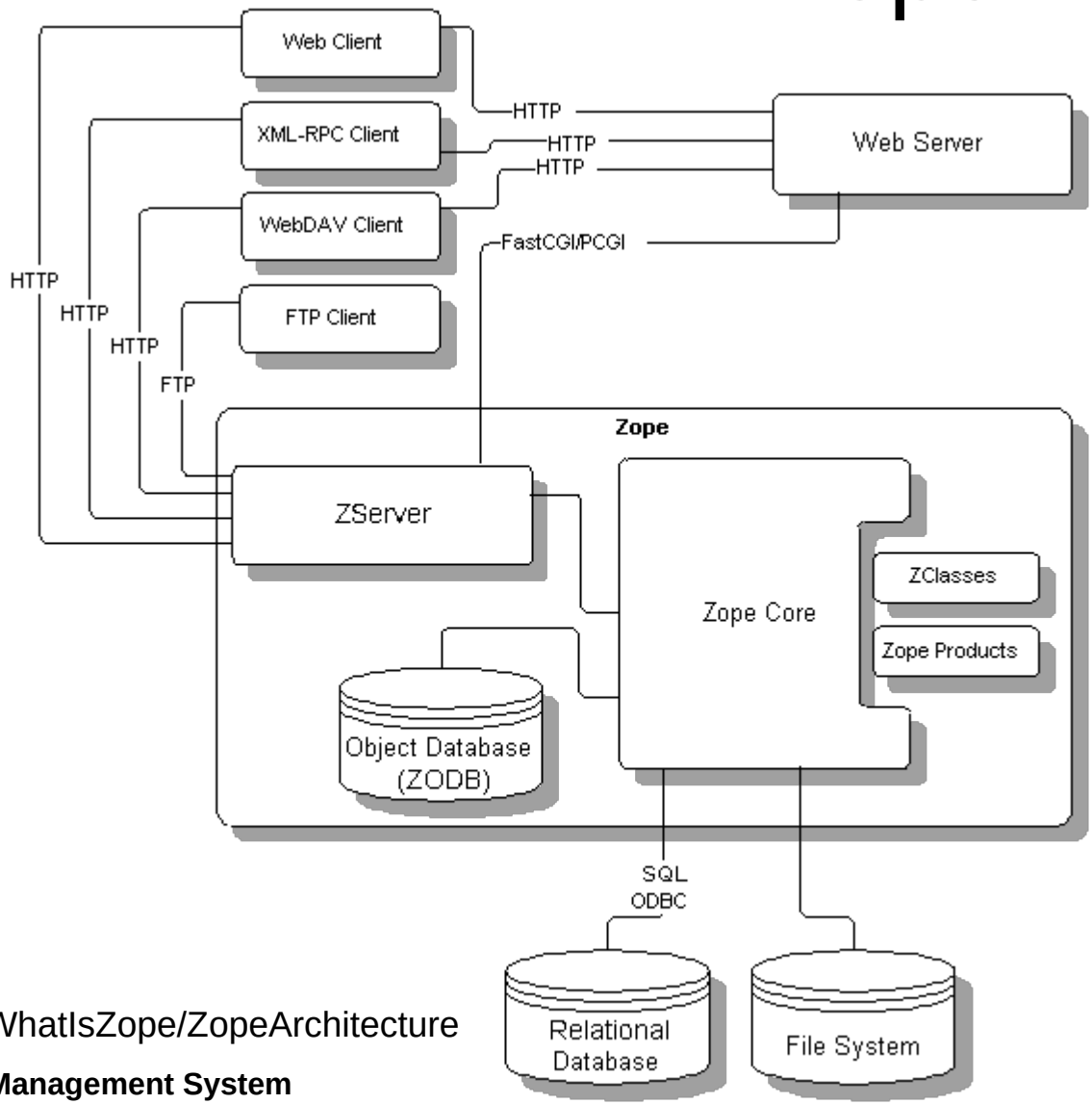
Practice 2 of 3

Practice	IDPR
Establish development and test environments	Two servers on the unclassified network will mirror servers on the classified network. Of the two servers, one is dedicated to production mirroring and one is dedicated to development mirroring.
Select software development methodology	Iterative development
Understand software design to the extent known	Yes
Understand software quality assurance	Yes
Understand risk analysis and mitigation	Yes
Understand software testing	Yes
Conduct walkthroughs	Close interaction with software developer will include walkthroughs at every milestone, as well as for the final product delivery.
Document subcontractor interfaces	Yes
Understand collaborations	Yes
Review and update requirements	Yes

Practice 3 of 3

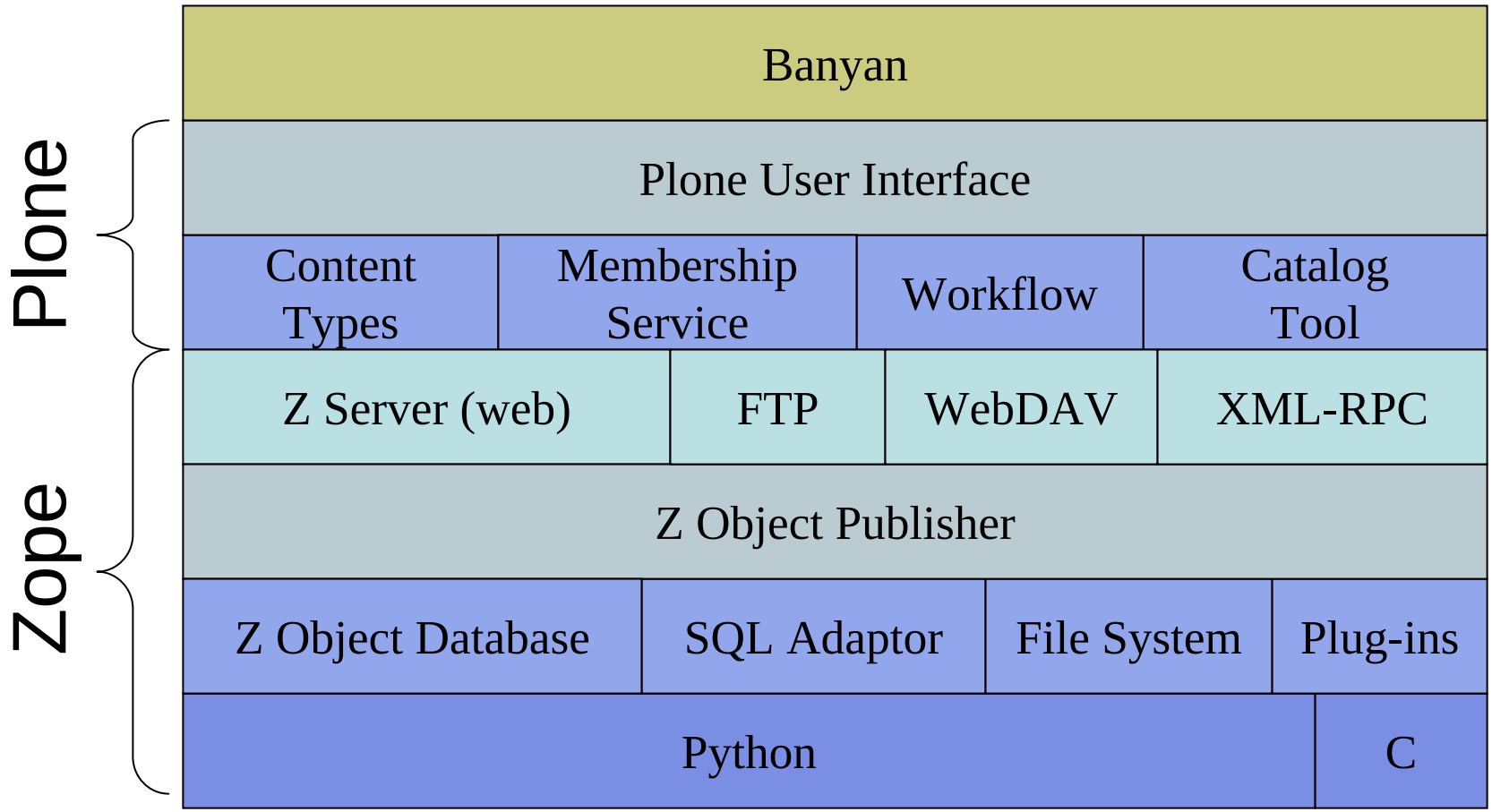
Practice	IDPR
Understand software configuration	Yes
Create informal disaster recovery strategy	Yes
Understand software integration	Yes
Conduct ad hoc acceptance testing	Yes
Understand feedback and status reporting	Yes
Provide maintenance support as needed	IDPR is an ADAPT-funded project. ADAPT philosophy is to turn maintenance and operation of systems over to the Program upon completion. We are actively attempting to broaden the user community to assure a sustainable user base.
Create release notes as needed	Yes

Banyan is based on Zope



http://www.zope.org/WhatIsZope/ZopeArchitecture

Zope/Plone/Banyan Stack



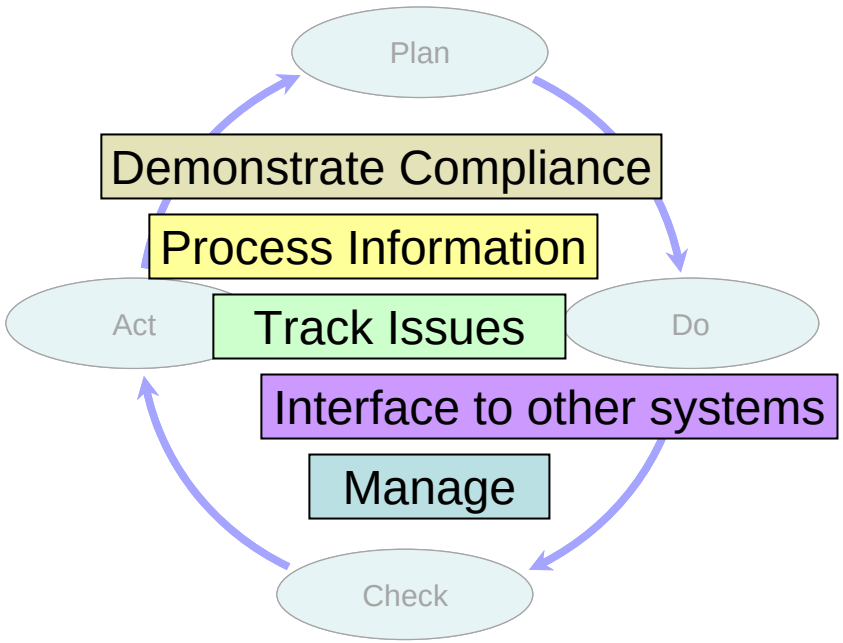
Important Attributes

- Web based, multi-user, collaborative tool
- Highly configurable

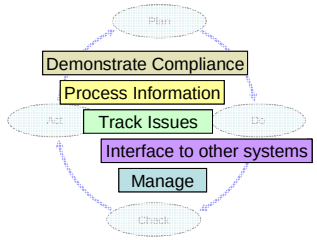
  Applicable to a wide range of problem domains
- Integrative technology
- Scalable
- Affords layered visibility

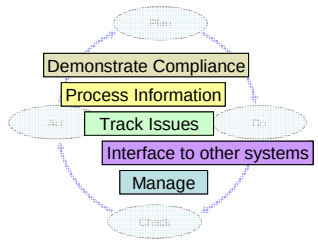
- **Rapid, hassle-free deployment**
- **Incremental integration with other systems**

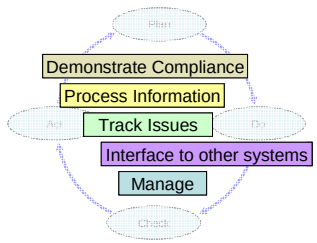
Banyan is part of a larger toolset

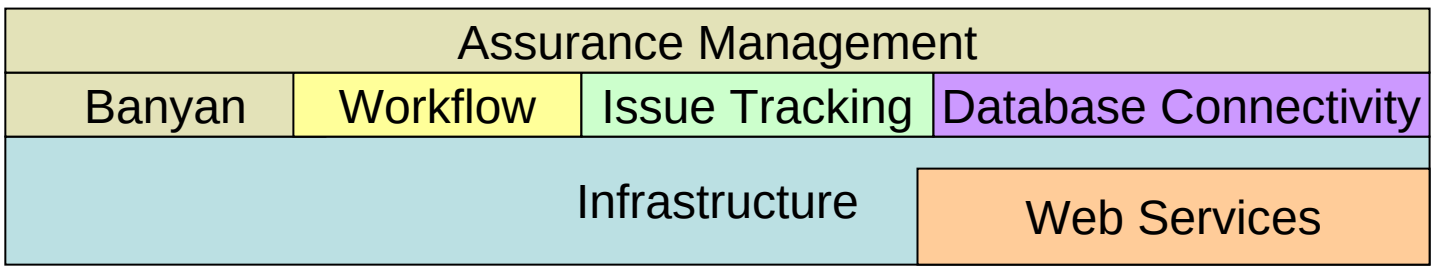


Distributed Workflow









Summary

- Proven track record
- Easily adapted
- Scaleable, integrative technology
- Easy to deploy and use
- Released as Open Source
- Looking for partners

The power of collaborative, clear thinking