HOW WE STAND NOW

- OK For Small Projects, Not So Good For Large Projects
- Not Good For Addressing Iterative Nature Of Requirements Resolution & Implementation (Mostly Based On Waterfall)
- Does Not Address Complexity Issues Of Requirements Stabilization (Based On Functional Decomposition)
- Does Not Explicitly Address Reuse Opportunities
- Does Not Help With People Shortages

NEED TO DEFINE AND AUTOMATE IMPROVED SOFTWARE ENGINEERING PROCESSES
REUSE AND PROTOTYPING - TWO SIDES OF THE SAME COIN

- Reuse Library Parts Are Used To Generate Good Approximations To Desired Solutions, i.e., Prototypes
- Rapid Prototype Composition Implies Use Of Pre-existent Parts, I.E., Reusable Parts
  - Prototype Quality Depends On Fit Of The Available Parts
  - The Parts Will Often Require Some Adaptation
  - As The Set of Parts Available Becomes Richer The Prototypes Will Better Approximate Acceptable Pieces of Final Systems
Big Gains In Productivity Will Come From Reusing Fewer Larger Parts Or Assemblies Of Smaller Parts, Not From Many Unassembled Small Parts.

Productivity Gain vs Cost Is Acceptable If Assemblies Of Parts Are Reused Frequently.
SYNTHESIS MOTIVATED BY AND ORIENTED TOWARD

- Reuse: Exploit Similarities Across Systems
- Iteration: Feedback and Enhancement
- Composition and Adaptation: Using Standard Schemes, Parts, and Designs
- Specialists: Incorporate Expertise, and Facilitating and Coordinating
- Systems View: Engineering Process
- Applying Synthesis to "Synthesizer"
THREE MAJOR SYNTHESIS SUBPROCESSES

Domain Analysis

Application System Modeling Toolset

Domain Independent Toolsets

Reuse Libraries

Project Library & Work Areas

Scavenging or Re-Engineering

System Development & Evolution

Reusable Work Products
Single System Work Products
Developed Toolset
ESSENCE OF DOMAIN ANALYSIS

- Each application area must be analyzed and characterized by standard *designs* or *architectures* that capture the way that many systems in that area could reasonably be built.

- The application engineer must be able to state his needs in application terms and have those needs mapped appropriately to an instance of the standard design.

- The design instance can be realized by specification of a set of parts from a reuse library and a set of rules for combining those parts.
SYNTHESIS SUBPROCESS - SCAVENGING

- Many systems with software have portions amenable to adaptation for reuse.
- Scavenging these systems for reusable parts involves:
  - Extraction
  - Generalization
  - Standardization
  - Certification
  - Cataloging and storing in reuse libraries.
A MISSILE GUIDANCE SYNTHESIS PROTOTYPE TOOL

An example of the application of reuse, prototyping, and synthesis using a reuse library in a specific domain

- Based on U.S. Air Force "Common Ada Missile Packages" (CAMP) parts
- Initially demonstrates a longitudinal autopilot control system
- Aids understanding of the economics of reuse
A METHODOLOGY FOR PARTS SPECIFICATION AND MODEL ASSEMBLY IS EVOLVING

- Based On NRL Software Cost Reduction Methodology
  - Information Hiding Module Families
  - Abstract Interfaces

- Accommodates Ada Packaging And Tasking Concepts
  - Tasking Guidelines Evolved (ADARTS)

- Initial Guidebooks Written And In Use
PRODUCT SET 1A STRUCTURE

AAD Editor

Diagram

Graphical

Database

AutoLayout

Informational Filters

Printer

PDL File

PDL->DIANA

PDL Semantic

Analyzer

(RADC FE)

Ada Skeleton

Generator

PDL database

(DIANA)

RADC DIANA

Diagrammer
DYNAMICS ASSESSMENT TOOLSET COMPONENTS

- Symbols
- AML Compiler
- SPNDL Compiler
- Model
- SPNM Analyzer
- Results
- Results Displayer
- TO SPNDL
- Graphic Editor
- Filters
- Transitions
  - 1 immediate 1.0
  - 12 timed 0.0 2.0
  - area input
- Task producer is begin process 500; buffer, put, and producer;
UIS ARCHITECTURE

**Tool**

- User Interface Code
- Computation Code

**User Interface services**

- Widget Instantiation Calls
- Tool Computation Callbacks

**X Window System**

- Ada Bindings
- HP Widgets
- SPC Widgets
- X Toolkit
- X Library

- X Protocol Network Messages

- X Display Server
  
  (or Integrated Server for X and Native Window System)
THE LAYERED REPOSITORY CONCEPT

ONE FOR EACH MEMBER
COMPANY NETWORK
LOCATION

ONE FOR EACH PROJECT

ONE OR MORE FOR
EACH DATABASE LOCATION
CREATED BASED ON
PROJECT NEEDS
(COLLECTIONS CAN CONTAIN DATA OBJECTS
AND/OR OTHER COLLECTIONS)

ACTUAL DATA
OBJECTS.
INCLUDES BOTH RDBMS STORAGE AND COTS CM STORAGE

REPOSITORY

PROJECT LIBRARY

PARTITION

COLLECTION

DISTRIBUTED
DATA
LEVEL

INDIVIDUAL
DATABASE
LOCATION
LEVEL

OBJECT
LEVEL
TYPICAL PROJECT LIBRARY ACCESS

APPLICATIONS CODE
- DESIGN TOOL
- ASSESSMENT TOOL
- TRACEABILITY
- HARNESS TOOLS
- MC DEVELOPED TOOLS

TYPICAL COMMANDS
DATA OBJECT RELATED
- CREATE DATA OBJECT
- DELETE DATA OBJECT
- CHECK OUT DATA OBJECT BODY
- CHECK IN DATA OBJECT BODY
- GET ATTRIBUTE
- SET ATTRIBUTE
- GET CONTENTS LIST

RELATIONSHIP RELATED
- CREATE RELATIONSHIP
- DELETE RELATIONSHIP
- GET ATTRIBUTE
- SET ATTRIBUTE

UNIQUE ID RELATED
- PATHNAME TO UID
- RELATIONSHIP TO UID

QUERY RELATED
- FETCH BY ATTRIBUTE VALUE
- GET RELATIONSHIP TYPES
- GET RELATIONSHIPS

DMS CODE

COTS PRODUCTS

TOOL K

RDBMS

DAS/DMS

OBJECT BODIES

DYNAMIC SQL INTERFACE

ATTRIBUTES

RELATIONSHIPS

TAILORED CODE INTERFACE

SOFTWARE

HODUCMVNR

SOFTWARE PRODUCTIVITY
### Traceability Navigator 1

#### NAV - Traceability Navigator

Requirements objects in /xyz/abc/def/jkl

Subset: date > 880325, owner = Johnson

<table>
<thead>
<tr>
<th>Name</th>
<th>Rev</th>
<th>Para number</th>
<th>Type</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>jlljkljk</td>
<td>5</td>
<td>3.1.1.2</td>
<td>cap</td>
<td>12Sep88</td>
</tr>
<tr>
<td>jlljllajfsj</td>
<td>4</td>
<td>3.1.1.2.1</td>
<td>ln</td>
<td>21Sep88</td>
</tr>
<tr>
<td>jklojdfmk</td>
<td>5</td>
<td>3.1.1.2.2</td>
<td>out</td>
<td>21Sep88</td>
</tr>
<tr>
<td>djlojkmkn</td>
<td>2</td>
<td>3.1.1.2.3</td>
<td>cap</td>
<td>17Aug88</td>
</tr>
<tr>
<td>jlljllajk</td>
<td>2</td>
<td>3.1.1.3</td>
<td>cap</td>
<td></td>
</tr>
<tr>
<td>jlljllajfsj</td>
<td>8</td>
<td>3.1.1.3.1</td>
<td>ln</td>
<td></td>
</tr>
<tr>
<td>jklojdfmk</td>
<td>4</td>
<td>3.1.1.3.2</td>
<td>out</td>
<td></td>
</tr>
<tr>
<td>djlojkmkn</td>
<td>12</td>
<td>3.1.1.3.3</td>
<td>cap</td>
<td></td>
</tr>
<tr>
<td>jlljllajk</td>
<td>6</td>
<td>3.1.1.4</td>
<td>cap</td>
<td></td>
</tr>
<tr>
<td>jlljllajfsj</td>
<td>3</td>
<td>3.1.1.4.1</td>
<td>ln</td>
<td></td>
</tr>
</tbody>
</table>

17 total

Tracing relationships to a set of related objects

### NAV - Trace Relationships

Tracing from requirements objects in NAV window 1

**specify relationship type(s):**
- tested by
- implemented by
- derived from
- described in
- sfiksa jfjadfo

**specify scope of tracing**

/lv23 ccms

GO
IMPACT ANALYSIS TOOLSET OVERVIEW

Modification Request → Person → Initial Impact Set → Impact Analysis Tool → Estimated Impact Set

OBJECT CHANGE LIST: List of impacted objects to examine

- Person
- Perform Change?
  - Yes → Impact Analysis Tool
  - No → Forget it!

Verify that person inspected/changed all objects in the change list

Impacts on Project Schedule
CANDIDATE USER INTERFACE FOR RLT

X-WINDOW System

Tool Display Windows

Types of Life Cycle Objects
- Documentation
- Source Code
- Requirements
- Design
- Executable code
- Miscellaneous

LCO Information Display
General Attributes
- Part ID: 1087
- Name: OFF Structure
- Description: The F15 OFF structure ...

Selection Criteria
- Classification
- LCO Types
- Attributes
- Relationships

SOFTWARE PRODUCTIVITY