Systems and Proposal Engineering Company

Knowledge-Based Analysis and Design (KBAD)

A methodology for rapid, cost-effective system engineering and architecture development

Presented October 22, 2008

SPEC™
Systems and Proposal Engineering Company

Overview of presentation

- Why yet another “methodology?”
- What is KBAD?
- What theory underlies KBAD?
- What kind of tools work with KBAD?
- What process does KBAD implement?
- What kind of people do we need to execute KBAD?
- How do we move from drawing pictures to building a knowledgebase?

Why Yet Another Methodology?

- **We have the DoD Architecture Framework …**
  - But DoDAF isn't a methodology, its just a description of necessary products
- **We have UML …**
  - But UML is only a software engineering technique. You have to come up with the process and tools for implementing it
- **We now have SysML …**
  - But SysML is just another technique and still needs more definition to create complete, executable designs
- **What's missing?**
  - A complete, coherent technique, process, and tool set that results in a knowledge base that can be used for full lifecycle decision making

SPEC Systems and Proposal Engineering Company

Knowledge-Based Analysis and Design

- **KBAD combines system engineering and program management disciplines to enable the development of a knowledgebase that can enable cost-effective decision making**
- **KBAD spans the acquisition lifecycle enabling support for design, development, integration, test, operations and sustainment**
- **KBAD focuses on using a variety of techniques and tools, brought together in a common database using special software to migrate data between tools**
- **The KBAD process links the technique and tools together in an executable, cost-effective way to support decision making at all levels**

KBAD reduces costs and increases speed of delivery by simplifying the data captured and focusing on the analyses needed for design. The result: a knowledge-base for decision making.

What makes up KBAD?

- **Technique**
  - Modified Model-Based System Engineering (MBSE)
- **Process**
  - SPEC's Middle-Out Process for Architecture Development and System Engineering
- **Tools**
  - A variety of COTS tools tailored to the MBSE modifications and special needs of DoDAF
- **People**
  - Trained, experienced professionals who bring a wealth of different backgrounds and knowledge in architecture, system engineering, modeling & simulation, physics, computer science, test & evaluation, operations & support

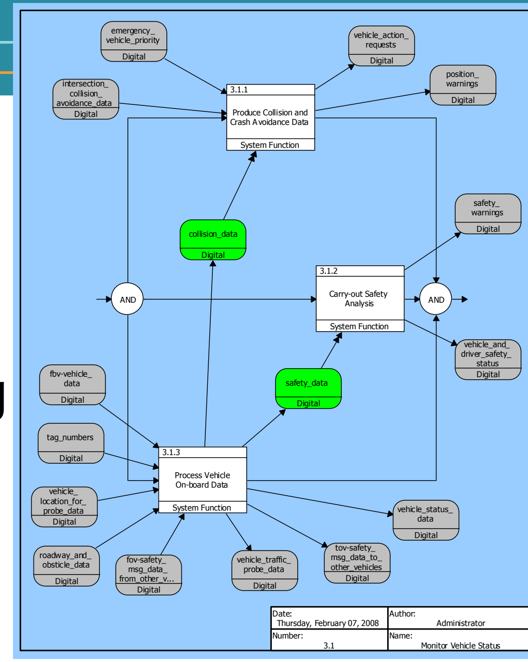
KBAD was developed over the past 15 years and brings lessons learned from those years of experience.

The technique: refined MBSE

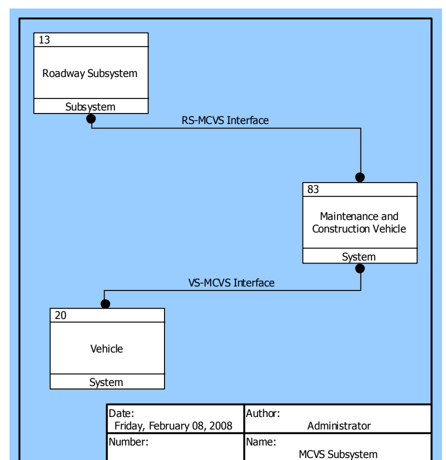
- Various forms of model-based system engineering have been developed
- SPEC uses the one developed by TRW in the late 1960s, which has been successfully used since then
- SPEC has refined this technique by simplifying the information collected (entities, relationships and attributes) and adding a number of key elements missing from the original development

MBSE Models

1. Logical architecture (behavior) model
   - Functional sequencing
   - Data flow and size
   - Resource model
   - Evolution in time



2. Physical architecture (asset) model
   - Interface definition (bandwidth and latency)
   - Actions allocated to Assets
   - Data allocated to interfaces



Models are based on language

Language Elements	English Equivalent	KBAD Schema Example
Element	Noun	• Statement • Action • Asset • ...
Relationship	Verb	• Statement is the <u>basis of</u> an Action • An Action is <u>performed by</u> an Asset • ...
Attribute	Adjective	• Description • Type (e.g., Operational Activity is a type of Action • ...
Attribute of Relationship	Adverb	• <u>amount</u> of Resource consumed by an Action • <u>acquire available</u> (hold partial) Resource for Action • ...
Structure Enables Executability	Graphics/ Drawings	• Graphic Views: Behavior, Hierarchies, Physical Block

We modified Vitech’s schema

KBAD Element	CORE Elements	Rationale
Action	Function/Operational Activity	Provide overall class for actions
Artifact	Document	Recognized not just documents
Asset	Component/Operational Element	Provide overall class for assets
Characteristic	type of Requirement	Way to capture metrics and other characteristics of an element
Cost	attribute of Component	Broadens capture of costs
Input/Output	Item/Operational Information	Clearer name
Issue	Issue	Same
Link	Link/Needline	Provide overall class for transmission
Location	none	Captures geolocation information
Risk	Risk	Same
Statement	type of Requirement	Clearer name
Time	attribute of Function	Broadens capture of times

The goal was to simplify and clarify the language.

SPEC Systems and Proposal Engineering Company

We related all the KBAD schema elements

	Action	Cost	Characteristic	Artifact	Asset	Input/Output	Link	Statement	Issue	Risk	Time	Location	*CORE Equivalent*	*DoDAF Equivalent*
Action	decomposed by	incurs	specified by	documented by	performed by utilizes	inputs outputs triggered by	-	based on	generates	resolves	occurs	located at	*Function*	Operational Activity/ System Function
Cost	incurred by	decomposed by	specified by	documented by	incurred by	incurred by	incurred by	based on	generates	incurred by	occurs	located at	*New*	N/A
Characteristic	specifies	specifies	decomposed by	documented by	specifies	specifies	specifies	based on	generates	causes	occurs	located at	*New*	N/A
Artifact	documents	documents	documents	decomposed by	documents	documents	documents	source of	generates	causes	occurs	located at	*Document*	N/A
Asset	performs utilized by	incurs	specified by	documented by	decomposed by	-	connected by	based on	generates	causes	occurs	located at	*Component*	Operational Node/ System Node
Input/Output	input to output from triggers	incurs	specified by	documented by	-	decomposed by	transferred by	based on	generates	causes	occurs	located at	*Item*	Operational Information/Data
Link	-	incurs	specified by	documented by	connects	transfers	decomposed by	based on	generates	causes	occurs	located at	*Link*	Needline/Interface
Statement	basis of	basis of	basis of	stated in	basis of	basis of	basis of	decomposed by	generates	causes	occurs	located at	*Requirement*	N/A
Issue	generated by	generated by	generated by	documented by	generated by	generated by	generated by	generated by	decomposed by	causes	occurs	located at	*Issue*	N/A
Risk	caused by resolved by	incurs	caused by	documented by	caused by	caused by	caused by	caused by	caused by	decomposed by	occurs	located at	*Risk*	N/A
Time	occurred by	occurred by	occurred by	occurred by	occurred by	occurred by	occurred by	occurred by	occurred by	occurred by	decomposed by	located at	*New*	N/A
Location	locates	locates	locates	locates	locates	locates	locates	locates	locates	locates	occurs	decomposed by	*New*	N/A

Reduced number of elements from 21* to 12, while adding time, location and cost

**CORE's DoDAF schema*

SPEC Systems and Proposal Engineering Company

A key attribute – *type*

- **We added a "type" attribute to all classes**
- **Each "type" attribute contains different designators for the parent class**
- **Examples:**
  - Assets can have types that include:
    - Operational Node, System, Component, Resource, Subsystem, System of Systems, Component, …
  - Actions can have types that include:
    - Operational Activity, System Function, Task, Mission, …
- **You can expand these lists to characterize anything in that class**
- **When we display the element, we use the type**

Using the type attribute we reduce the complexity and ease changes in perspective from requirements to implementation.

SPEC Systems and Proposal Engineering Company

Benefits of the KBAD Schema

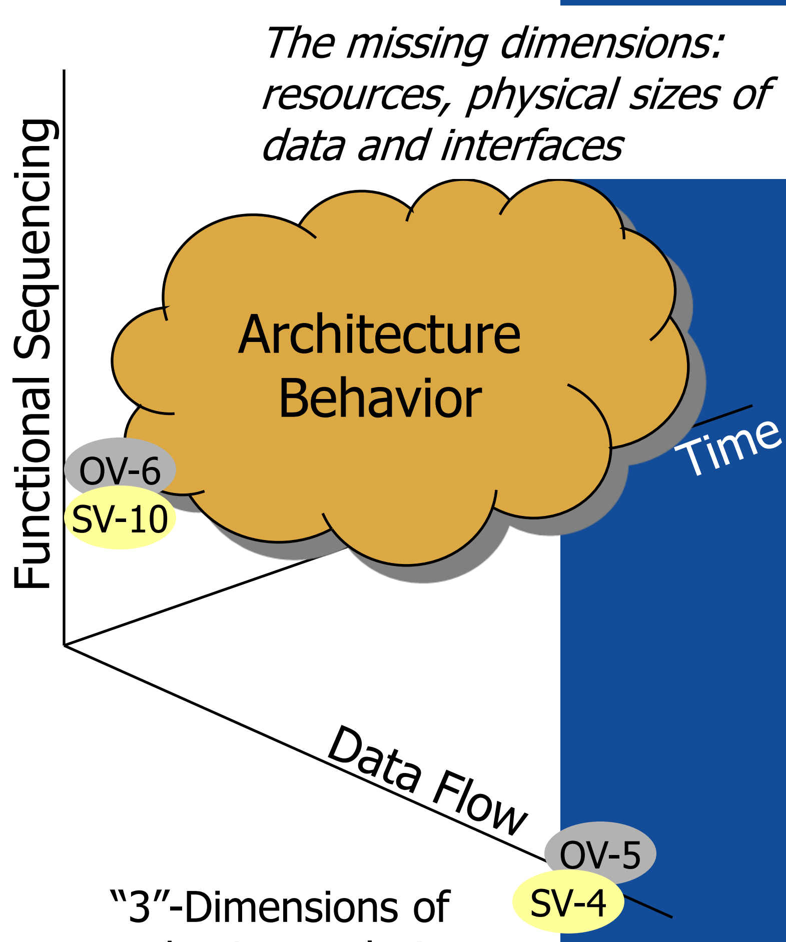
- **Reducing the number of primary data elements means less complexity for analysts to deal with**
  - Less complexity enables quicker capture and presentation of the information for analysis and decision making
- **Covers programmatic, as well as technical, elements of information**
  - Enables the trade off between cost, schedule and performance necessary for good design and decision making
- **Eliminates overlap between similar data elements**
  - Reduces potential for duplication of information which cuts the time and cost of data gathering

The result is a more cost-effective means for describing an architecture or system design.

SPEC Systems and Proposal Engineering Company

MBSE Describes Behavior

- **Typical data/activity modeling only works in the data dimension (e.g. IDEF0 or Data Flow Diagrams)**
- **For simple systems with sequential flow, this is sufficient**
- **However, for more complex systems, which all architecture are, it can be very misleading**
- **We need to be able to predict how system will behave**



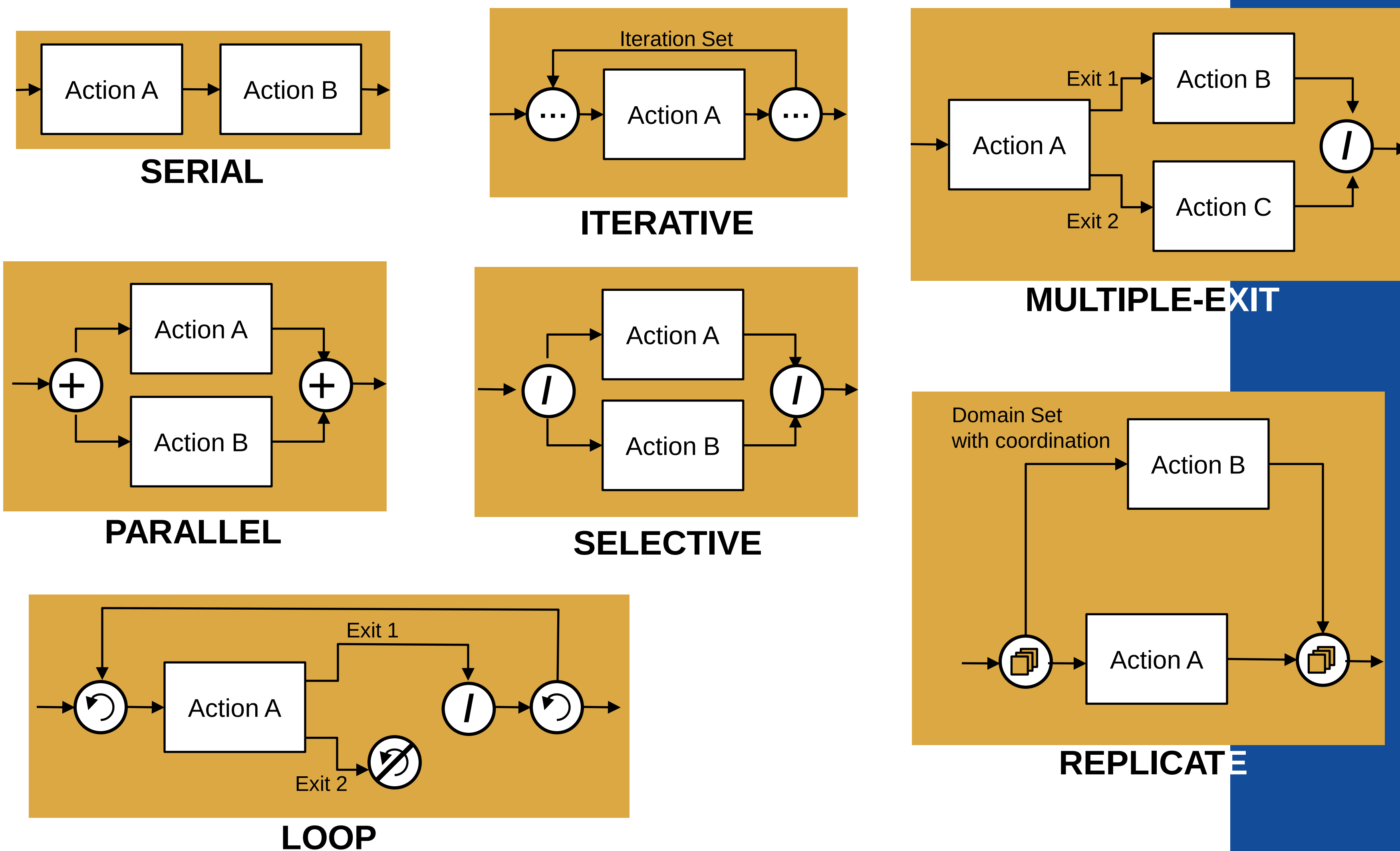
"3"-Dimensions of Behavior Analysis

Why is sequencing important?

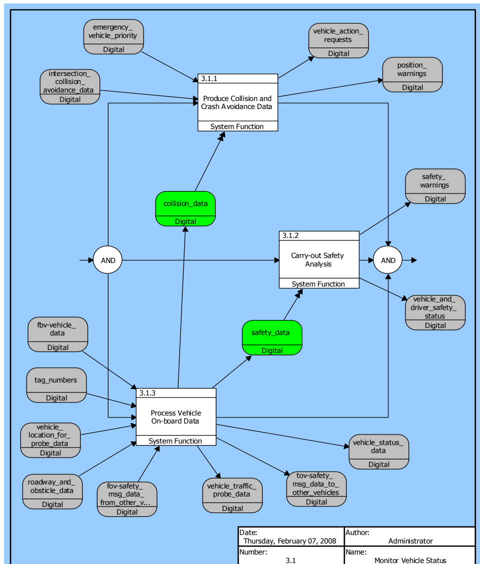
- **In software the mantra is: data, data, data**
  - Why? Because a tremendous amount of software programming has to do with input/output, hence the need to understand the data very well
  - The functional sequencing for individual software modules is relatively simple and many algorithms exist for complex methods (e.g., sorting algorithms)
- **In architecture development (or system engineering or business process modeling …) sequencing is actually more important than the data**
  - We want to know how the data affects the functional sequencing – we call these triggers
  - We want to control the behavior to avoid having significant failures
  - We also need sequencing for the human side

Hence the real answer is we need both if we are to develop systems and services with predictable behavior.

MBSE provide a robust set of constructs

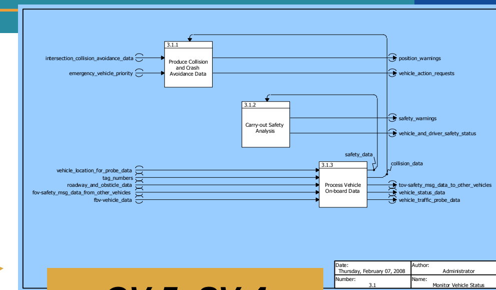


One diagram gives many products



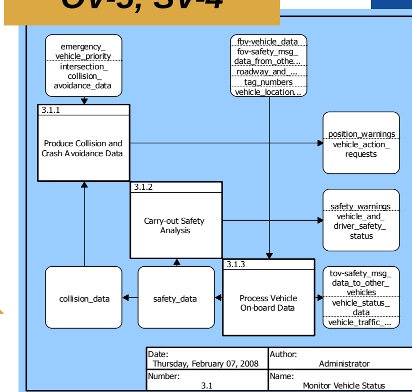
EFFBD: complete and executable

IDEF0: lacks constructs

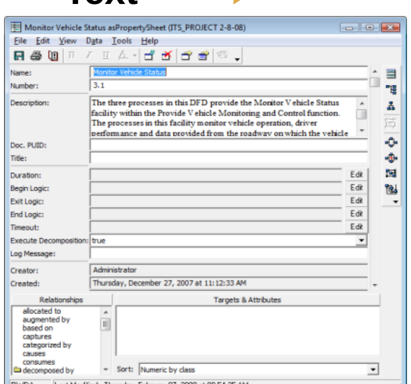


OV-5; SV-4

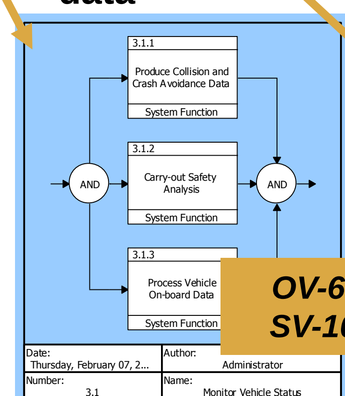
N2 Chart: lacks constructs



Text

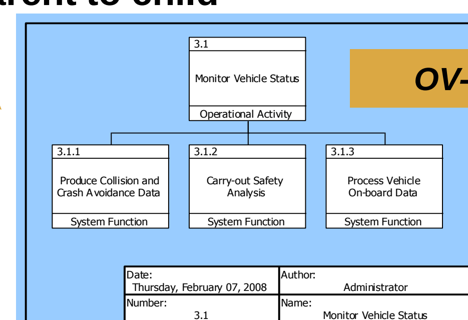


FFBD: lacks data



OV-6c; SV-10c

Hierarchy: only parent to child

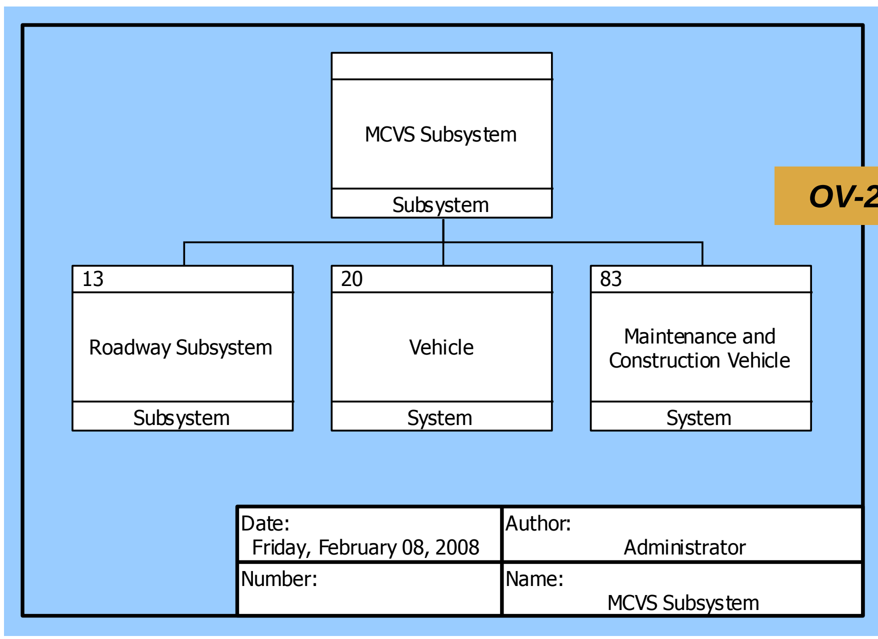


OV-5; SV-4

SPEC Systems and Proposal Engineering Company

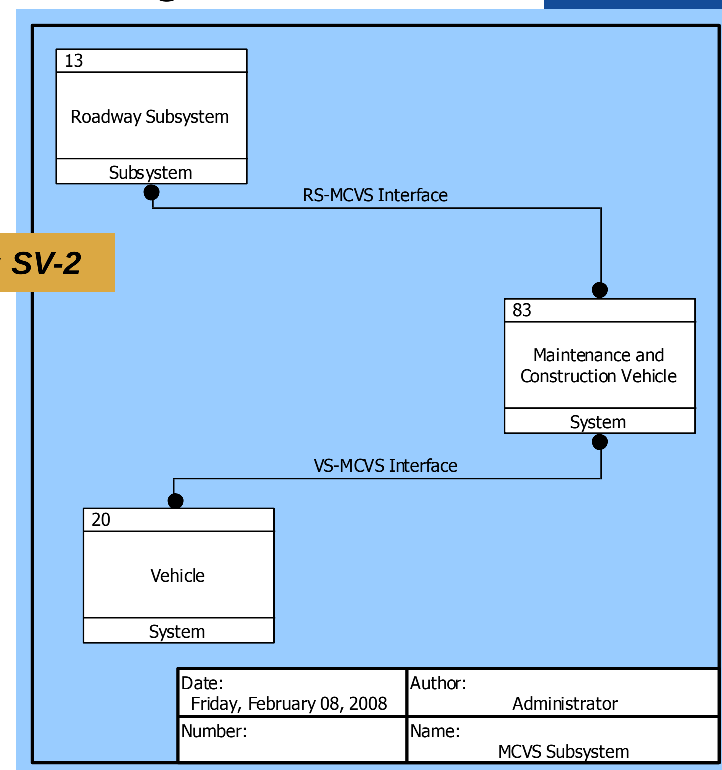
MBSE also diagrams the physical elements

Physical Hierarchy

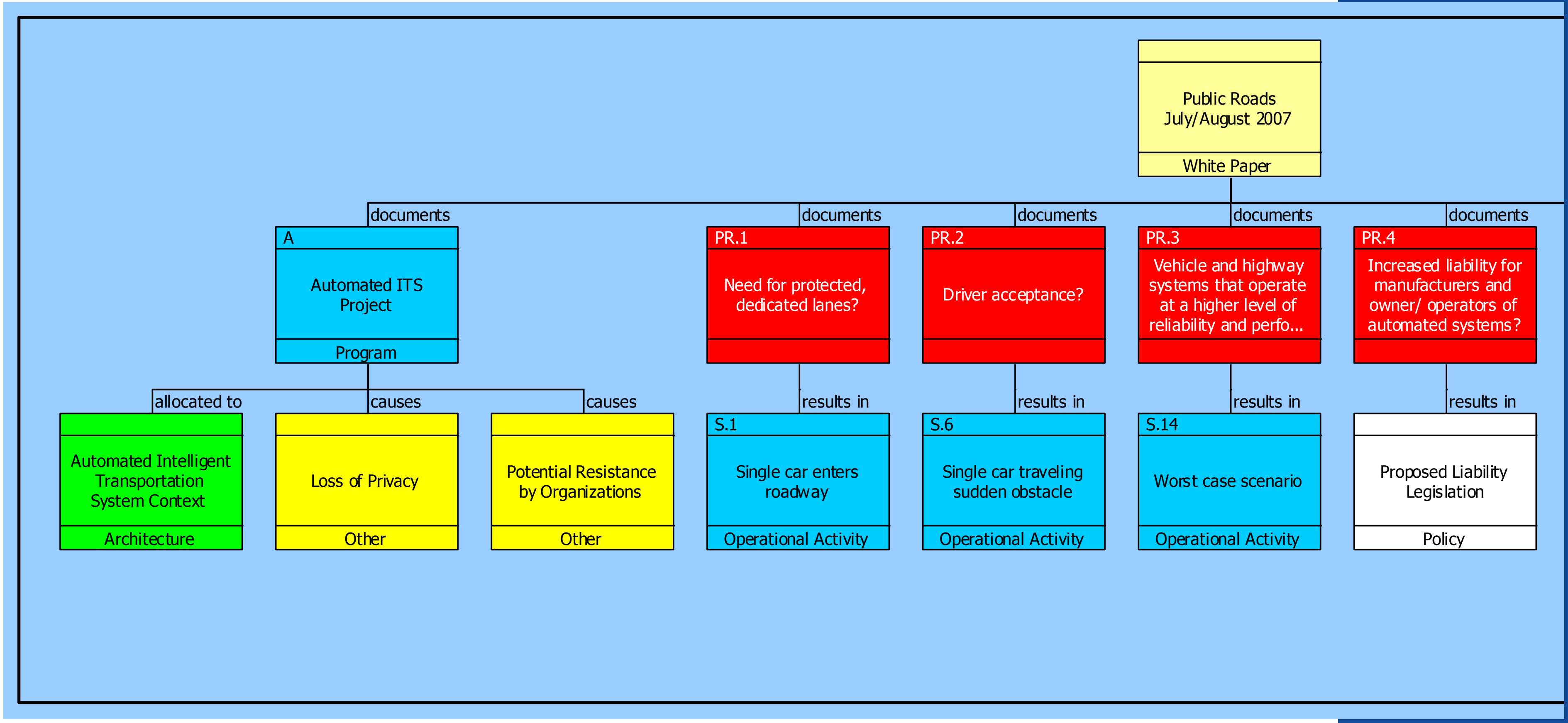


OV-2; SV-1; SV-2

Physical Block Diagram

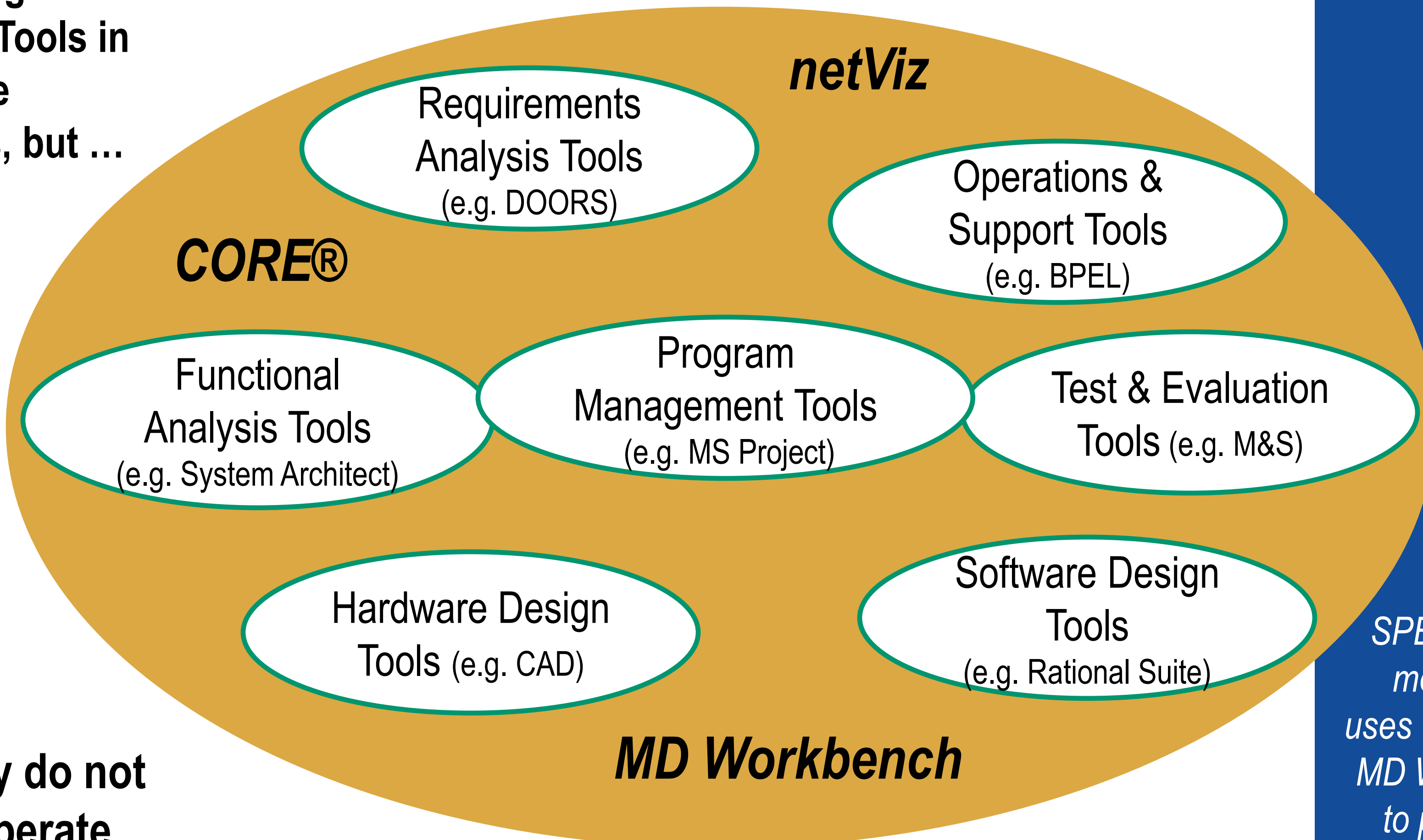


Traceability is a key to success



Tools support the technique and process

Most Programs Require Tools in All These Domains, but …

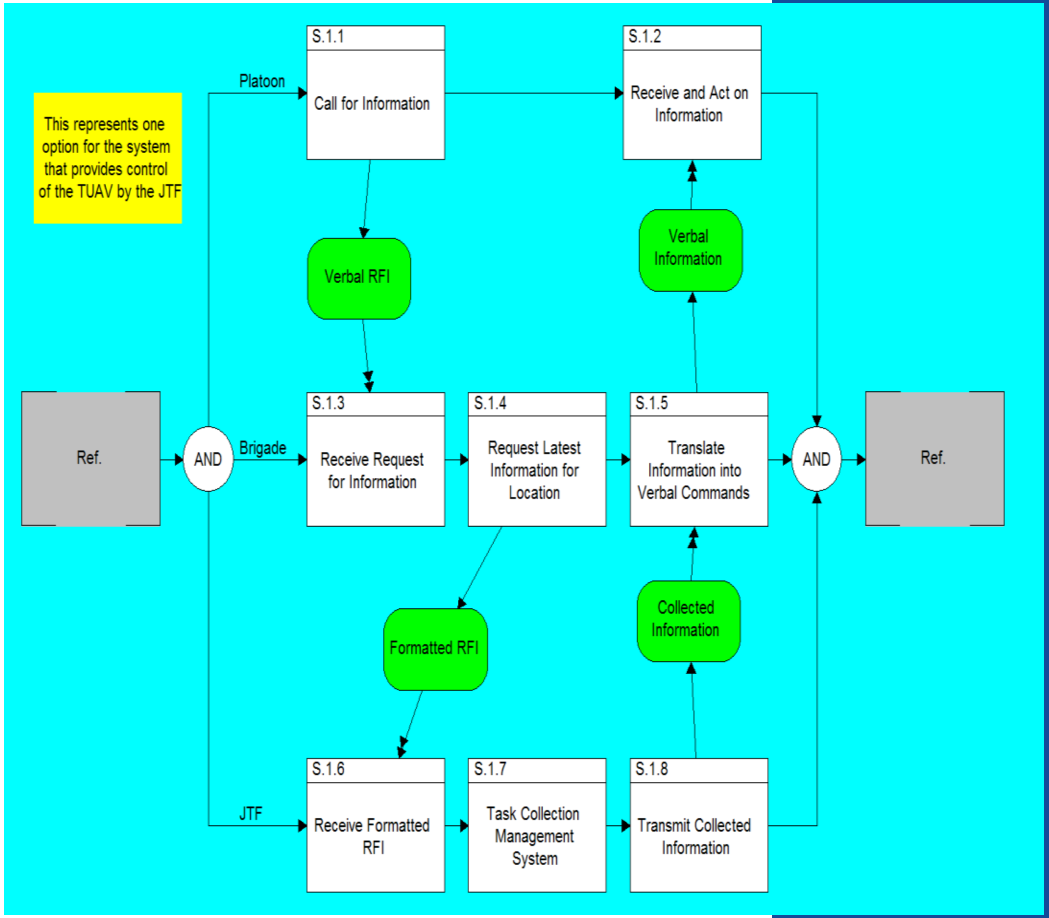


… they do not interoperate well together.

SPEC's KBAD methodology uses CORE and MD Workbench to provide the underlying tool interoperability.

Tools used: CORE

- CORE's system engineering tools maintain an integrated design repository that provides traceability between requirements, functional models and system design elements
- CORE's database schema may be modified to customize the tool to support customer needs and facilitate tool integration
- Executable diagrams
- Special schemas and reports
- Powerful scripting language for your own report generation



Version 5.1 released with updated schema and reports

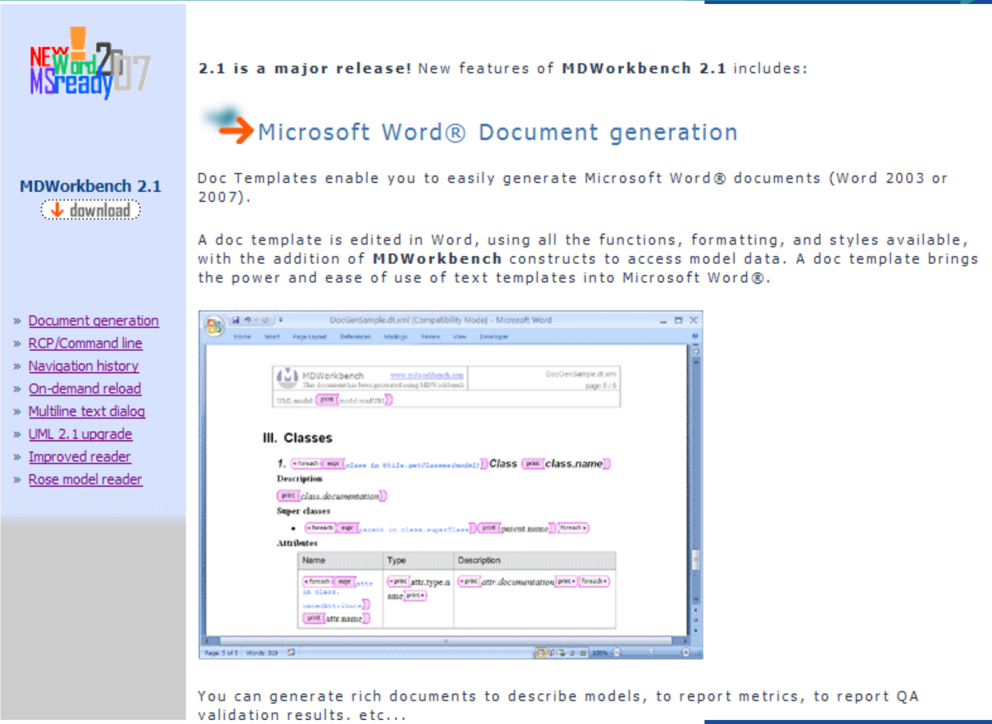
www.vitechcorp.com

SPEC Systems and Proposal Engineering Company

Tools used: MD Workbench

Eclipse-based IDE for code generation and model transformation, devoted to implementing MDA/MDE strategies. It provides:

- code generation (via text template engine and optionally Java)
- model manipulation through dedicated languages
- (imperative rules, declarative ATL modules to support QVT transformations, Java)
- model and metamodel management, including UML support
- customizable model connectors (XMI 1.0 to 2.1, XML, Hibernate, COM, etc.)

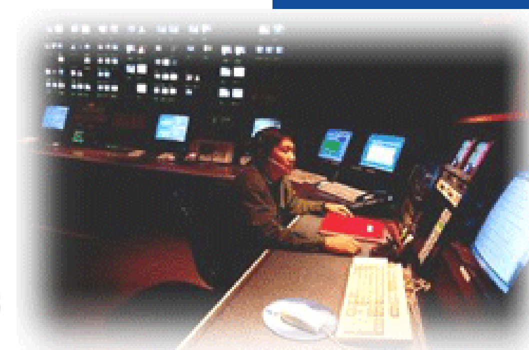


http://www.mdworkbench.com

A great way to move data between different tools.

Tools used: NetViz

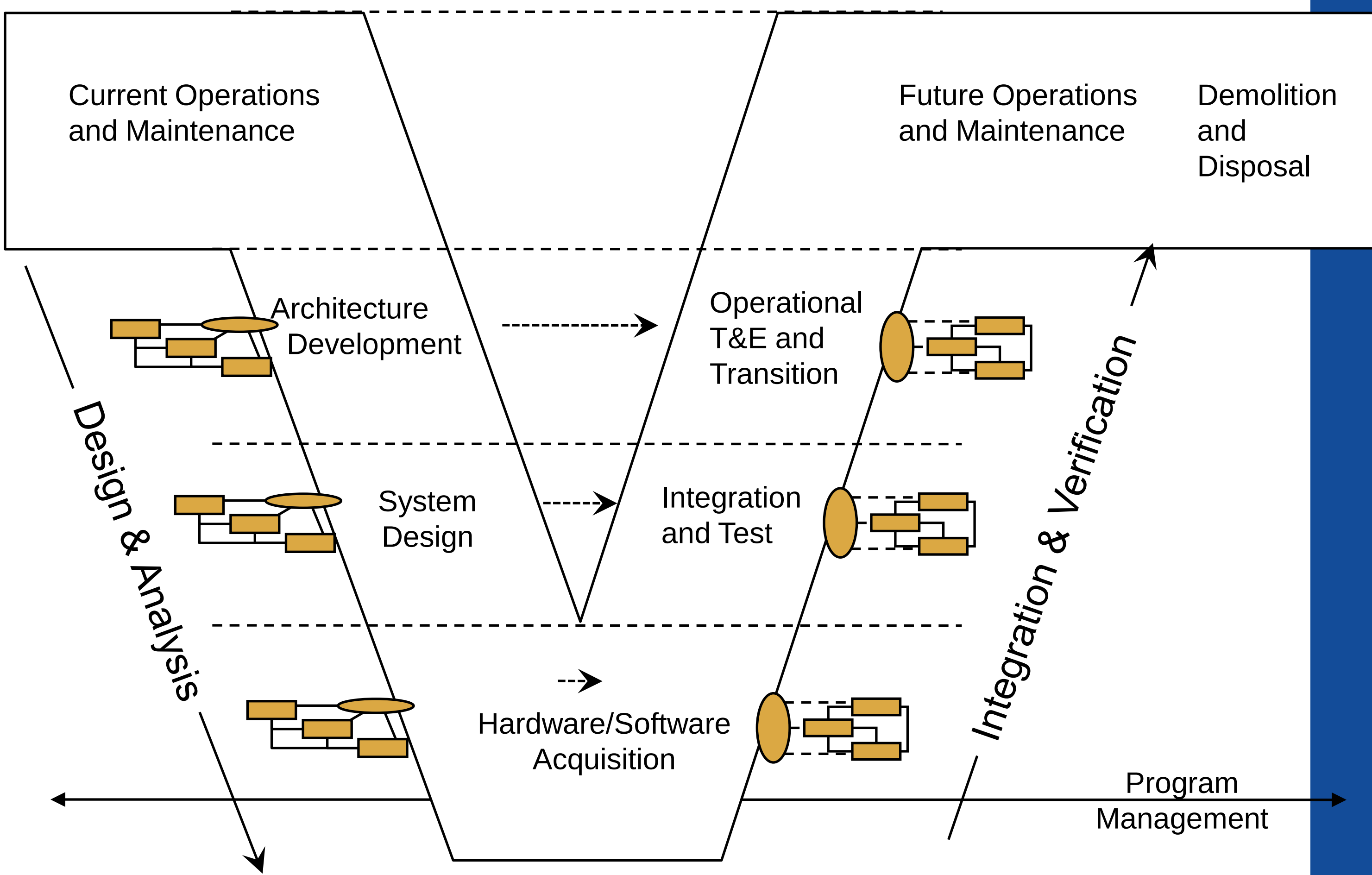
- **Personnel all over the world use netViz to graphically depict operational architectures and logistical scenarios**
- **With NetViz you can create the SV-1 and SV-2 diagrams, with its intuitive graphical workspace, drill down capability, and connectivity views**
- **You can use the data embedded in your netViz projects to create other critical elements of a comprehensive C4I documentation project, like OV-1s (Operational Concept Diagrams) and OV-3s (Information Exchange Matrices)**



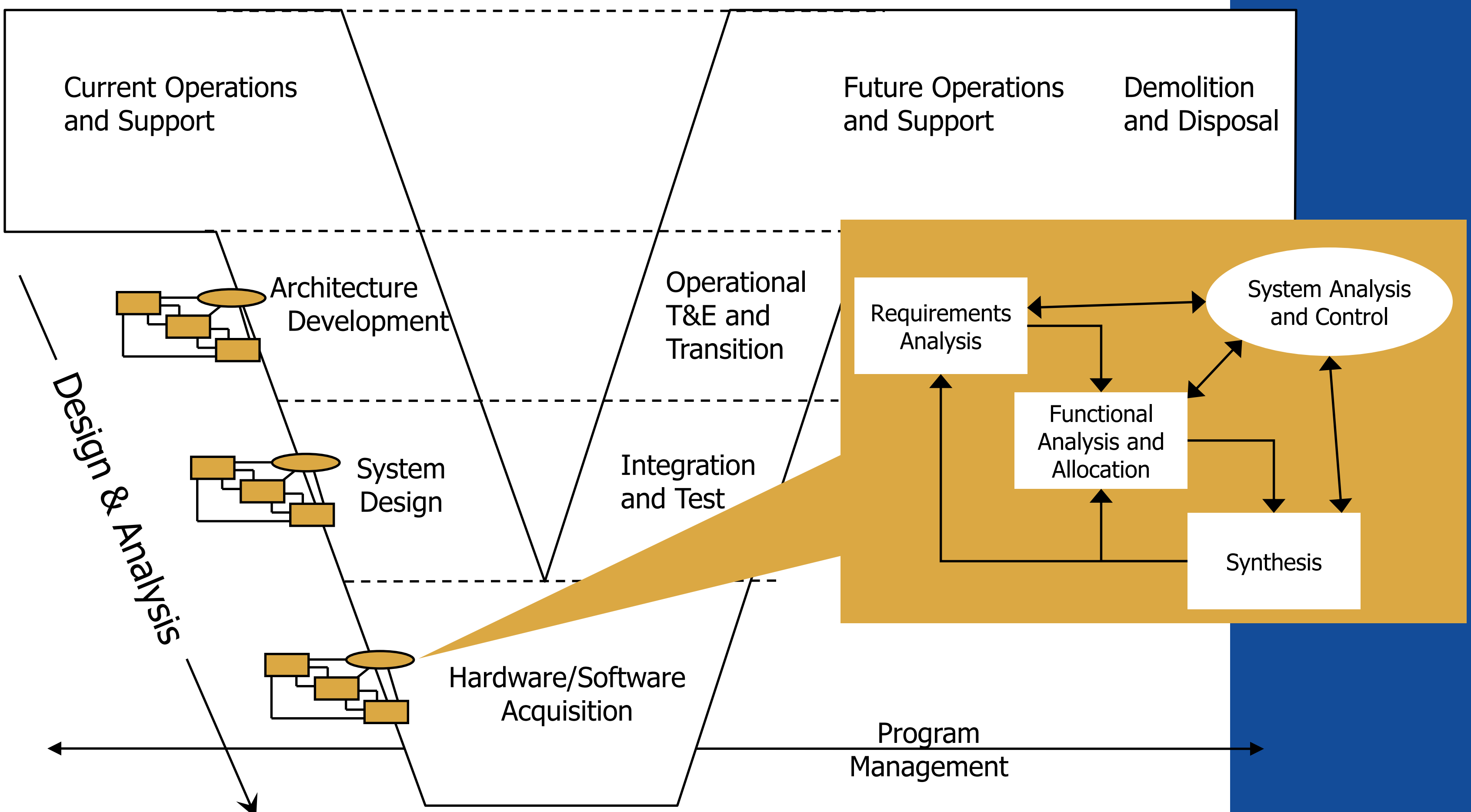
Version 7.1 released; Available in Client Server or Enterprise Web editions as well

www.netviz.com

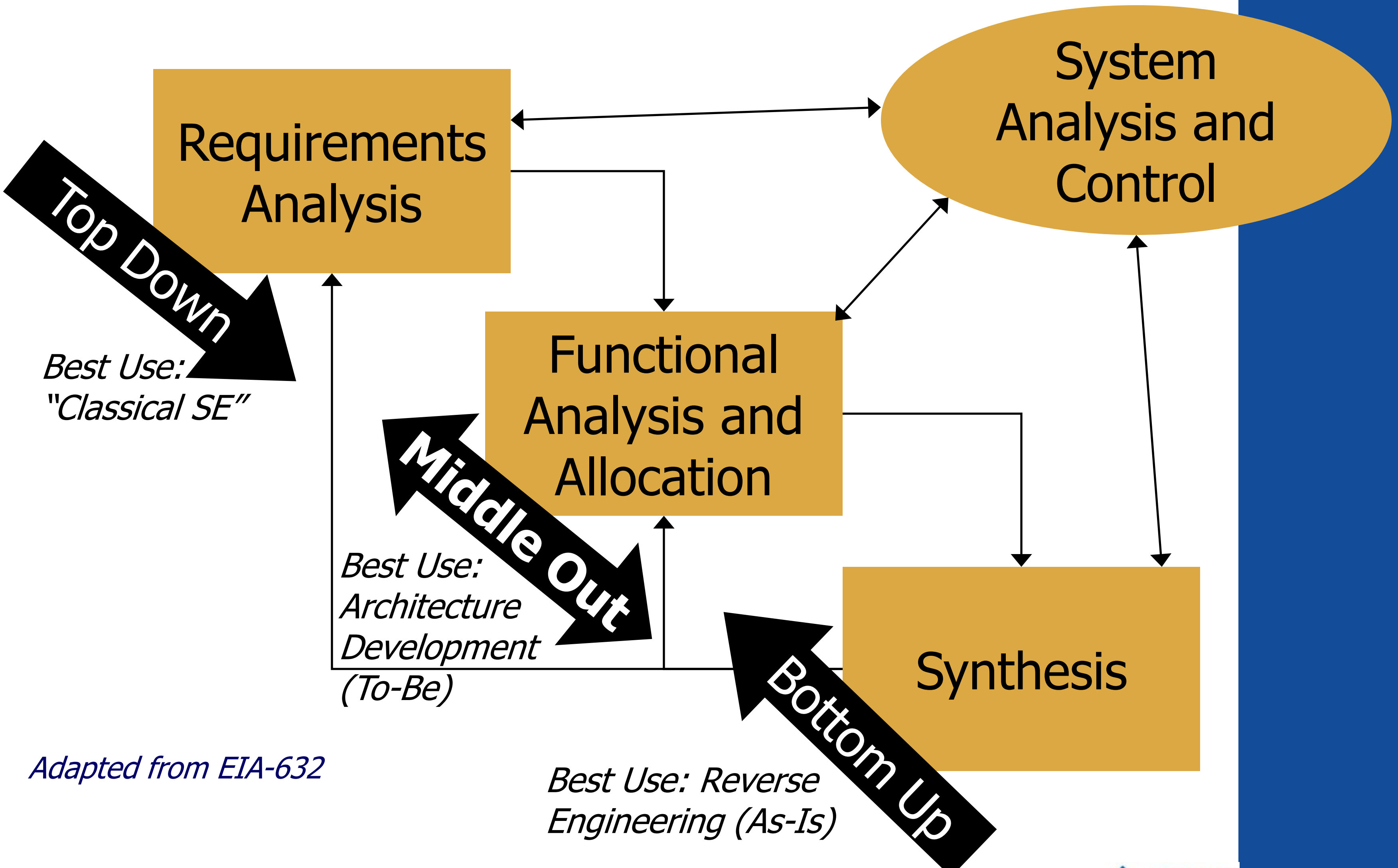
SPEC processes – full lifecycle



Design and analysis phase

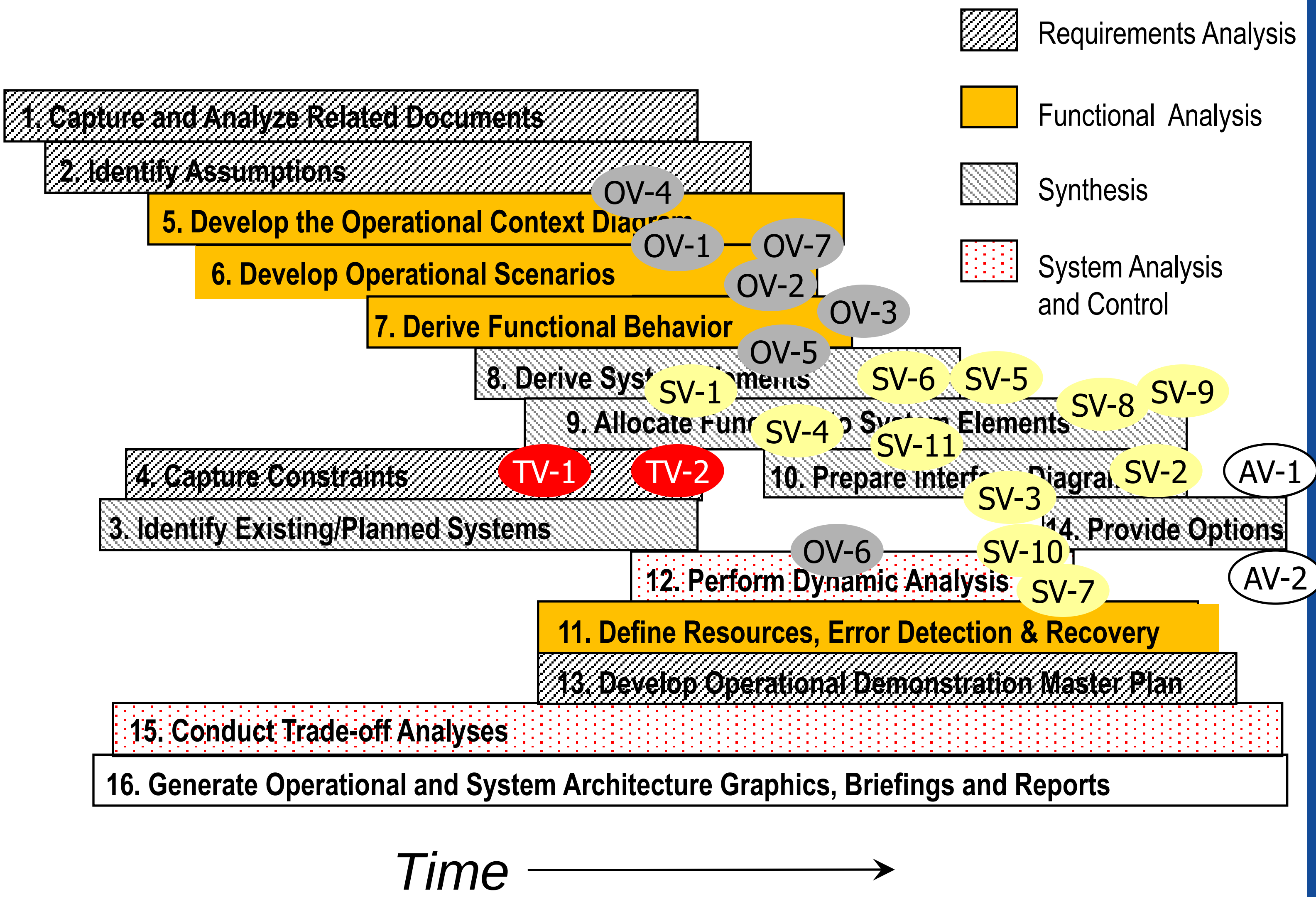


SPEC's middle-out process



Adapted from EIA-632

Middle-out timeline with products



The middle-out approach has been proven on a variety of projects.

People Considerations

- **Large teams make organization and focus on a vision very difficult**
- **You need people with a wide variety of skills and personalities**
  - Someone with vision
  - Someone who can perform the detailed system engineering
  - Someone who understands the domain
  - Someone familiar with the technique and tools
  - Someone who understands the process
- **They need to be trained as a team – including the government personnel**

How Do We Move from Drawing Pictures to Building a Knowledgebase?

- Apply a proven, model-based technique that results in executable diagrams
- Use a process that implements the technique
- Use industrial-strength system engineering tools
- Make sure the personnel who use the methodology have the proper knowledge, skills and abilities to implement the approach

Questions & Discussion