Systems & Software Technology Conference – 2007

Integrated Execution Threads

Dynamic System Definition

June 20th, 2007

Dr. John Forrest Harrell

The Aerospace Corporation

Integrated Execution Threads
Agenda

- A short definition and discussion of salient features
- Execution threads at the top level
- Execution threads at the bottom level
- Execution threads and links in the system hierarchy
- Advantages during design and development
- Advantages during test – verification and qualification
- Advantages after fielding – maintenance and modification
- Summary

Integrated Execution Threads
A Few Definitive Statements

- **An execution thread is a *sequence of discrete steps* within a system or system of systems that begins with an *external* initiating event, set of conditions or other impetus and concludes with an *external* event, set of conditions or other impact(s)**
- **The integrated execution threads (set of) document(s) covers *all* behavior of the system at its boundaries with the world around it**
  - Mission execution(s)
  - Error/anomaly detection and recovery
  - Maintenance, health and status, other non-mission
- **And *all* behavior for each subsystem, component and element at *their* boundaries with the environments around them**
- **The integrated execution threads document begins as a *prescription* for the anticipated development and finishes as a *description* of what was actually developed**

Integrated Execution Threads

At Each Level, All Threads Should First Be Listed in Categories

- List the nominal operational threads – how the system behaves as it executes its mission in the usual ways
- List the error response/recovery threads – how the system responds to off-nominal and anomalous events and conditions
- List the test and verification threads – any special execution threads needed to generate verification and qualification data
- List the maintenance threads – what the system has to do to keep itself healthy or to provide data to determine its health
- List the special threads – the launch certification thread, for instance, or the launch initiation thread

…But *DO NOT* forget to start…

…With the system the *customer* sees!!

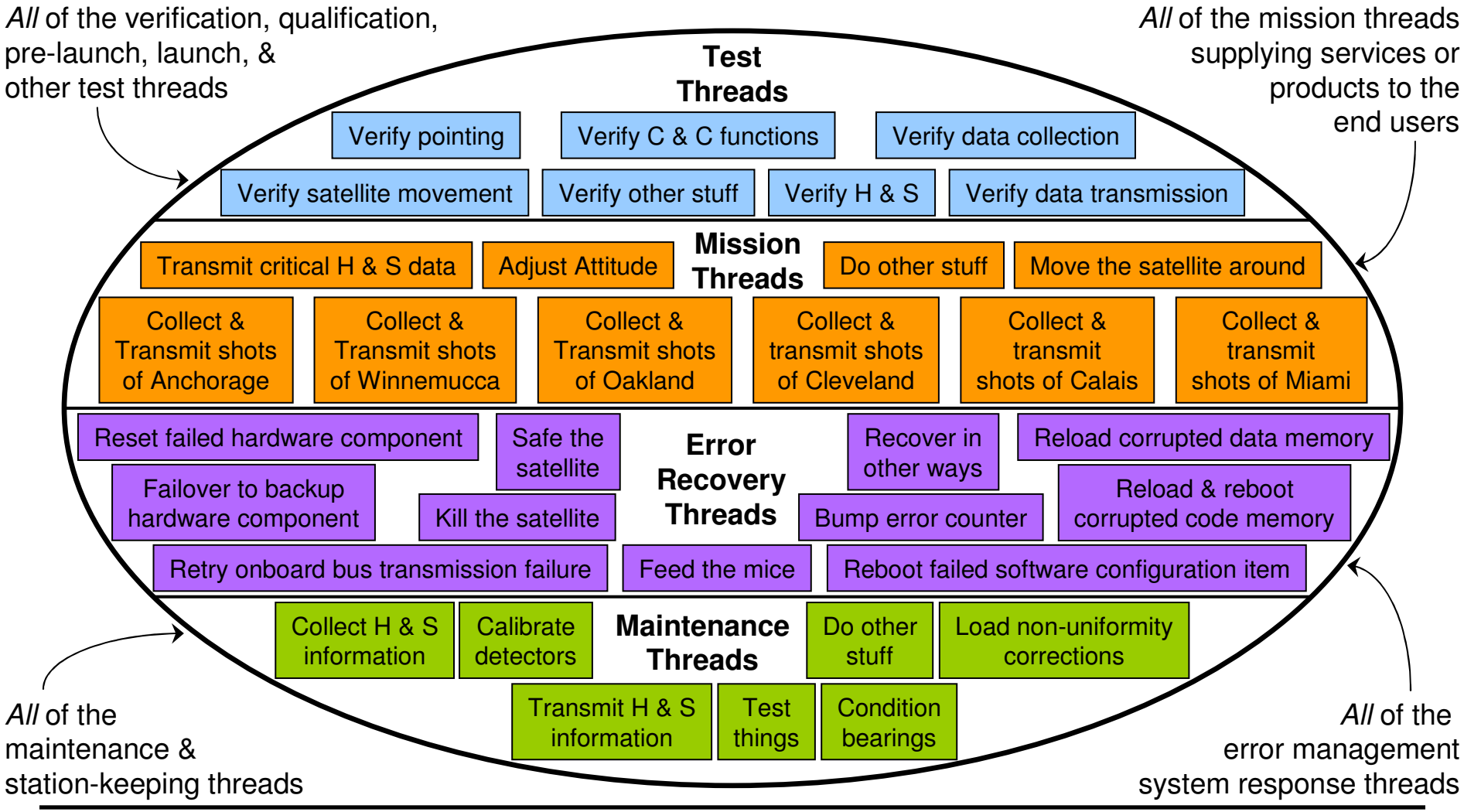
Integrated Execution Threads

The Top Level Is Not Just the Satellite, For Instance

- **The overall system (of systems, let's say) execution threads have to do with the information the customer wants – not the maintenance of the satellite, its ground stations and networks**
  - Infrared pictures of Calais in Maine on a regular basis for the Alcohol, Tobacco and Firearms people
  - Infrared pictures of Cleveland in Ohio on a regular basis for the Health and Human Services Department
  - Infrared pictures of Miami in Florida on a regular basis for the Immigration and Naturalization Service
  - Infrared pictures of Anchorage Alaska on a commanded basis for the Bureau of Land Management
  - Infrared pictures of Winnemucca in Nevada on a commanded basis for the Agriculture Department
  - Infrared pictures of Oakland in California on a commanded basis for the Justice Department

Integrated Execution Threads

Cover All System Behavior At Each Level (Now for Our Satellite)



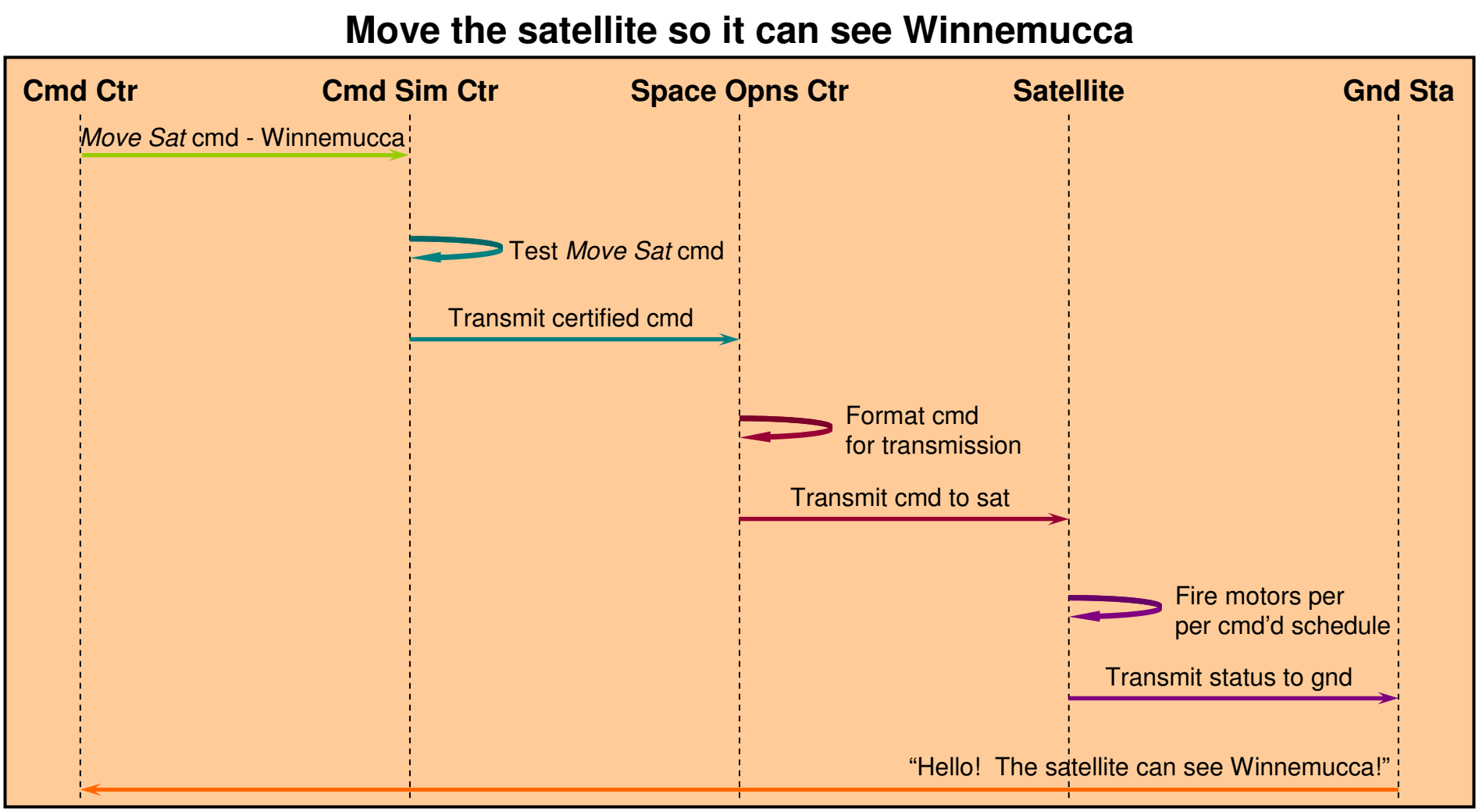
Integrated Execution Threads

A List of Satellite (Subsystem) Threads

#	Thread Name	Lower Threads
B.1	**Mission Threads**	
B.1.1	Move the satellite around	C.1.1, C.1.2, C.1.4
B.1.2	Adjust attitude	C.1.1, C.1.2,
B.1.3	Collect and transmit shots	C.1.1, C.1.2,
B.1.3.1	Of Anchorage	C.1.1, C.1.2,
B.1.3.2	Of Winnemucca	C.1.1, C.1.2
…	Of …	C.1.1, C.1.2, …
…	…	…
B.2	**Error Recovery Threads**	
B.2.1	Bump error counter	C.2.1
B.2.2	Reboot failed software item	C.2.1, C.2.2
B.2.3	Reset failed hardware component	C.2.1, C.2.3
…	…	…
B.2.n	Safe the satellite	C.2.1, C.2.12
…	…	…
…	…	…

Integrated Execution Threads

Diagram Sequences at All Levels – Even the Top One



Integrated Execution Threads

Write Down Each Step in Each Execution Sequence

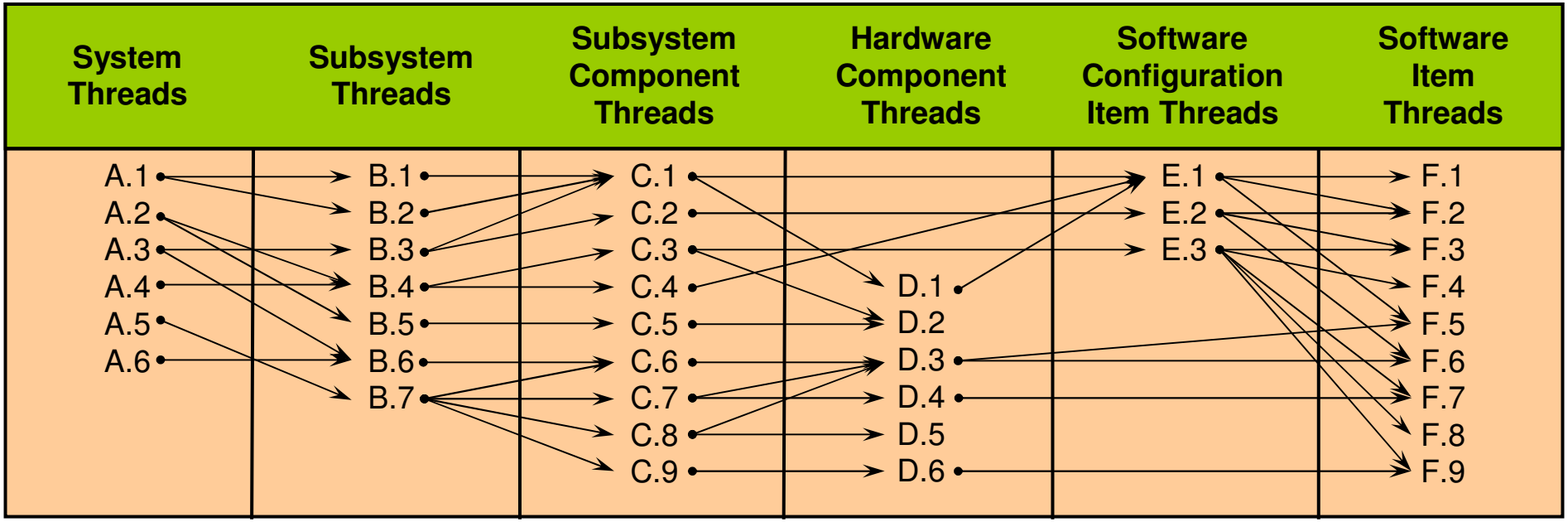
Step	Action	Lower Threads	Upper Threads	Action By	To/From
B.1.1	**Move the Satellite Around**				
-1	Send *Mov Sat* command w/ destination coordinates for pre-execution cert	C.1.1, C.1.2, C.1.3	A.1, A.2, A.3, A.4, A.5, A.6	Command Center	**To**: Sim Center
-2	Test *Mov Sat* cmd to verify/certify effective & non-detrimental execution	C.2.1, C.2.4, C.2.5	A.1, A.2, A.3, A.4, A.5, A.6	Simulation Center	
-3	Send certified *Mov Sat* command with specified destination for execution	C.2.2, C.2.3	A.1, A.2, A.3, A.4, A.5, A.6	Simulation Center	**To**: Space Opns Ctr
-4	Format command/firing schedule for transmission to satellite	C.3.1, C.3.5, C.3.6	A.1, A.2, A.3, A.4, A.5, A.6	Space Opns Ctr	
-5	Send command/firing schedule to satellite	C.3.2, C.3.3, C.3.4	A.1, A.2, A.3, A.4, A.5, A.6	Space Opns Ctr	**To**: Satellite
-6	Move satellite to specified destination coordinates	C.4.1, C.4.2, C.4.4	A.1, A.2, A.3, A.4, A.5, A.6	Satellite	
-7	Notify ground station of location of satellite	C.4.3	A.1, A.2, A.3, A.4, A.5, A.6	Satellite	**To**: Space Opns Ctr
-8	Notify command center that satellite has moved to commanded loc	C.3.7, C.3.8	A.1, A.2, A.3, A.4, A.5, A.6	Space Opns Ctr	**To**: Cmd Center

Integrated Execution Threads
Execution Threads at the Bottom Level

- **Execution threads at the bottom level involve three basic kinds of elements**
  - Hardware component behavior, including interfaces with processors and software, must *usually* be inferred from physical hardware specifications, programming interface specifications, and other such documents
  - Software configuration item behavior is *often* defined in software design documents and interface requirements specifications, as well as in final and software item qualification test threads (not necessarily execution threads, but close)
  - Software item behavior is defined in use cases for the individual items, *usually* including universal markup language diagrams
  - These things are usually *disjoint* representations of various parts of the overall detailed system design
  - Design artifacts explicating system behavior should be in *one* place and should be clarified with a unifying document of *some* kind – in an Integrated Execution Threads folder – that links to higher-level threads

Integrated Execution Threads

Execution Threads and Links in the System Hierarchy



- **Some systems are hierarchical, straightforward…and *simple*…**
- **Most, however, are more complicated…the *interesting* ones…**

So, one-to-one, one-to-many and many-to-one links
…are *probably* in there…somewhere…

Integrated Execution Threads
Advantages During Design & Development

- **An Integrated Execution Threads document or (more likely) folder provides unique value to the design & development of a system**
  - Requirements specifications and interface control documents *imply* defined system behaviors but rarely if ever *define* them
  - Behavioral changes in a system at any *given* level – from the Concept of Operations down to individual hardware components and software items – usually have impacts at *other* higher and lower levels
  - As requirements are allocated and derived the *design* of the system solidifies – and so should its *behavior*
  - Requirements changes that have an impact on system behavior at a *given* level can have an indirect effect on requirements at *other* levels because of multi-level changes in system behavior
  - System behavior is not merely inferred, but is defined, refined and clarified in *one* place for the initiate *and* the novice
  - Using simulation and prototyping following the integrated execution threads documentation, system behavior can be more easily modeled and demonstrated *early* and *often* to keep the program *sold* to the customer(s)

Integrated Execution Threads

Advantages During Test – Verification & Qualification

- **An Integrated Execution Threads document or folder provides unique value for the verification and test of a system**
  - What to do for software integration testing within hardware components
  - What to do for integration testing for hardware and software components
  - What to do for subsystem integration testing
  - What to do for subsystem software item qualification testing
  - What to do for system integration testing
  - What to do for system software item qualification testing
  - What to do for system final or functional qualification testing
  - What to do for pre-launch final system checkout
- **Wherever these tests focus on the behavior of the system, they benefit from documentation of the behavior sequences under test *and* from their relative place in the hierarchy of behaviors inherent in the overall system design**

Integrated Execution Threads

Advantages After Fielding – Maintenance & Modification

- **An Integrated Execution Threads document or folder provides unique value after fielding a system for follow-on maintenance and modification programs**
  - Maintenance engineers new to the system have *one* place to go to learn about its structure and its operation
  - Analysis of anomalies can begin with the *observed* system behavior at a given level and proceed according to the system design upward and/or downward through related behaviors in other levels
  - Modifications to the system can be specified and costed on the basis of a coherent and structured understanding of the change's impact on the behavior of the *whole* system
  - As the procedures governing the use of the fielded system change, they can be documented and presented in the *same* place that also elucidates what *lower-level* (or maybe even *upper-level*) behavior they affect

Integrated Execution Threads
Summary

- ***All* successful system development efforts have used *some* of this approach**
  - System behavior has been *defined* at all or most levels – just not *integrated* among them
  - Informal, *tacit* knowledge of integrated system behavior in the heads of (usually) old people guides the concluding integration and verification steps
- **An Integrated Execution Threads document or folder provides *unique* value**
  - For the design and development of a system
  - For the verification and test of a system
  - After fielding a system for follow-on maintenance and modification efforts
- **This approach seeks to make *explicit* and *structured* what has in the past been *tacit* and *informal* knowledge of integrated system behavior**