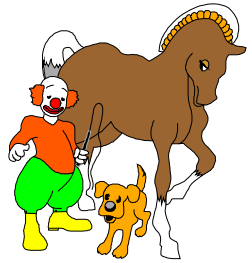
Specification-Driven Prototyping for Architecting Dependability **(and Dependable Architecting)**

Workshop on Architecting Dependable Systems
Orlando, Florida
May 25, 2002

Dennis B. Mulcare - Consultant
cefsm@ellijay.com
(706) 276-2497

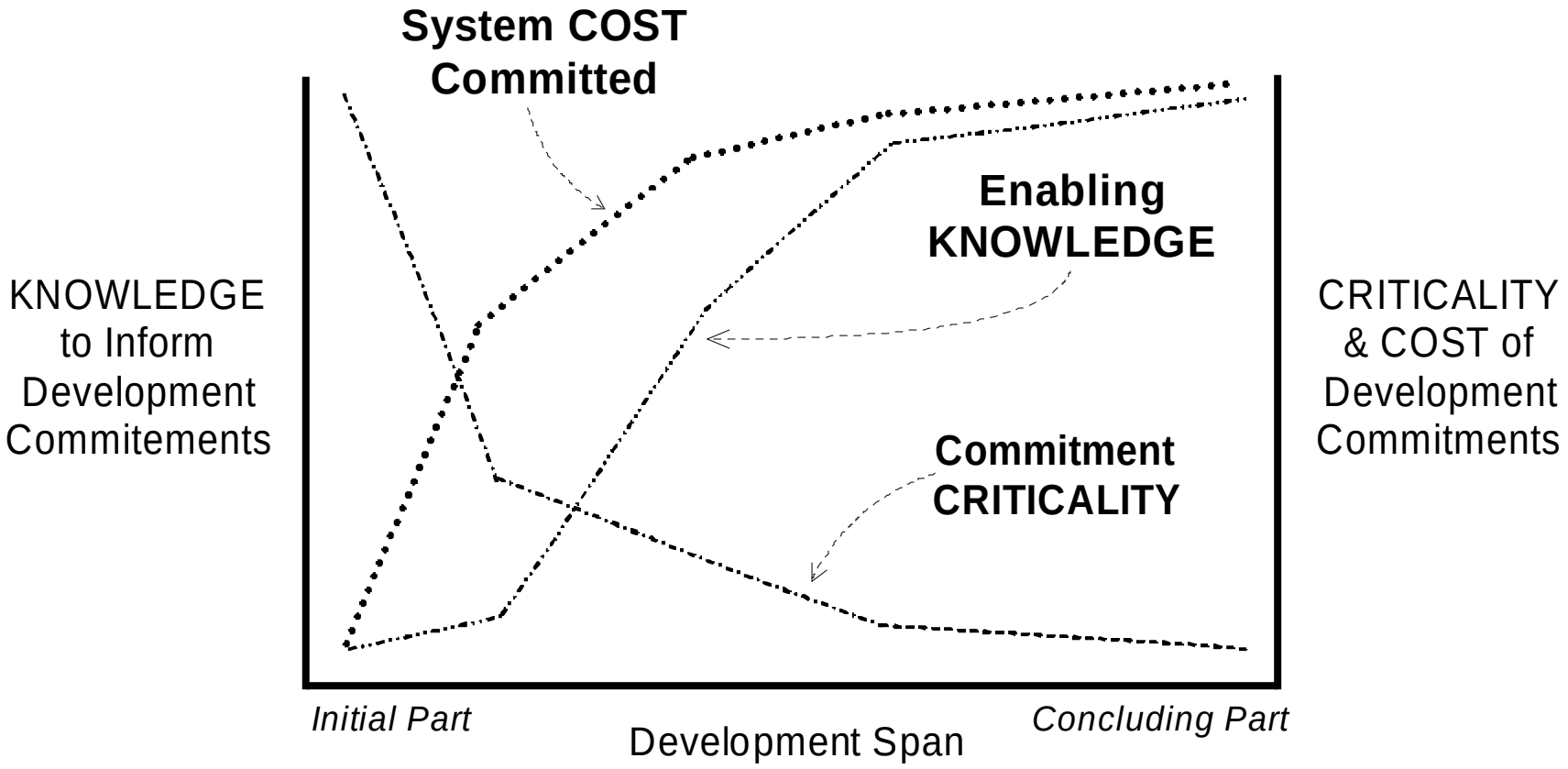
WADS-Centric FOCUS

(Hard real-time safety-/mission-critical context)



- **ARCHITECTING**
  - Accommodating *essential* problem complexity
  - Specification-driven prototyping *discipline & exactitude*
- **DEPENDABLE**
  - Dependable methodology a *prerequisite* to dependable products
  - Strong association with *extra-functional* properties
- **SYSTEMS**
  - *Systemic* nature of dependability properties
  - Primary leverage in system-level *infrastructure*

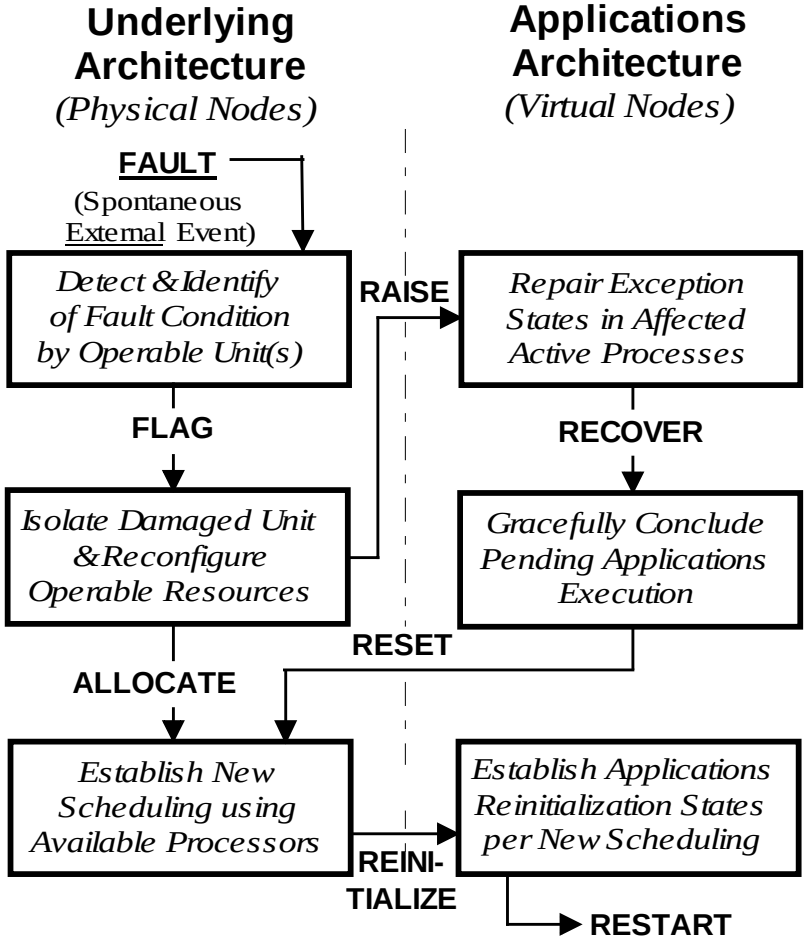
Architecting CIRCUMSTANCES



Vital Prototyping
OUTPUTS

- **PROBLEM EXPLORATION**
  - Timely focus on “hard” problems
  - Problem complexity: scope, subtleties & variations
  - Complete & coherent architectural solution
- **ANALYSIS SUPPORT**
  - Empirical data
  - Assumption validation & enforcement
  - Testing of the analytically intractable
  - Worst-case scenarios
- **PRODUCT SPECIFICATION CONTENT**
  - Component count & sizing
  - Quantitative parameters & tolerances
  - Global concurrency logic

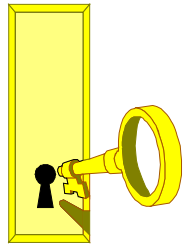
Prototype EXECUTION
(Fault Handling)



Apt & *Dependable* METHODOLOGY

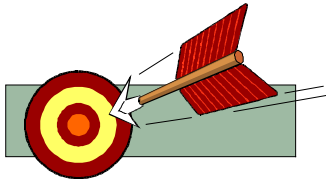
- **Infrastructure-Applications Architecture Partitioning**
- **Complementary Analysis-Simulation**
- **Specification-Driven Prototyping**
- **Higher-Level Statecharts**
- **Precise Dependability-Related Specification Content**

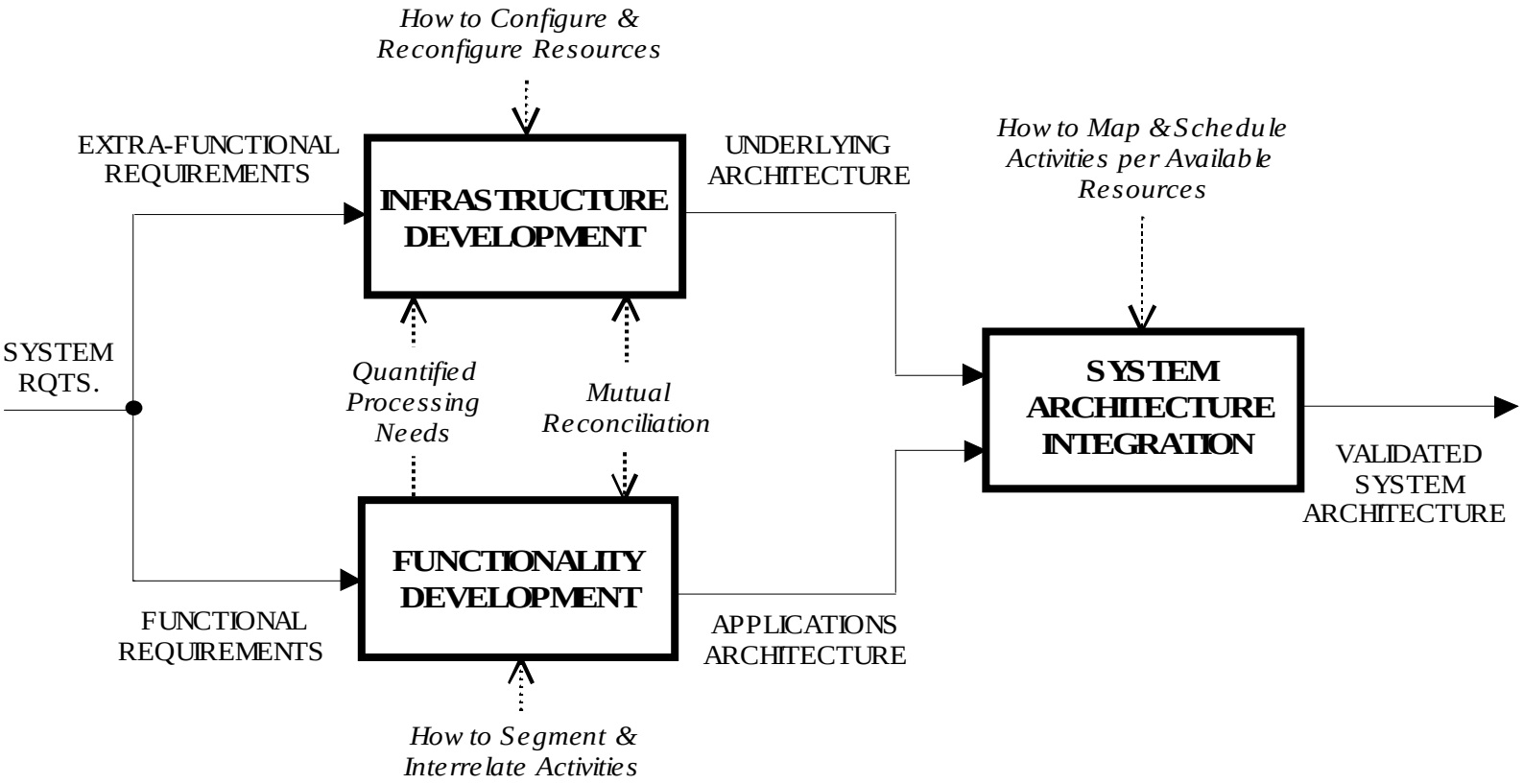
Applications versus Infrastructure ARCHITECTURES



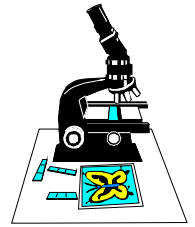
APPLICATIONS ARCHITECTURE	INFRASTRUCTURE ARCHITECTURE
Functional Requirements	Extra-functional Requirements
System Services	System Properties ("Ilities")
What Kind(s) of Service	How Well Service is Supported
Operational MODE	System STATE
Functional Performance	Infrastructure Performance
Shades of Grey Criteria	GO/NO-GO Criteria

Architecting Methodology
PARTITIONING





Complementary ANALYSIS-SIMULATION

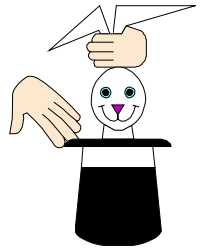


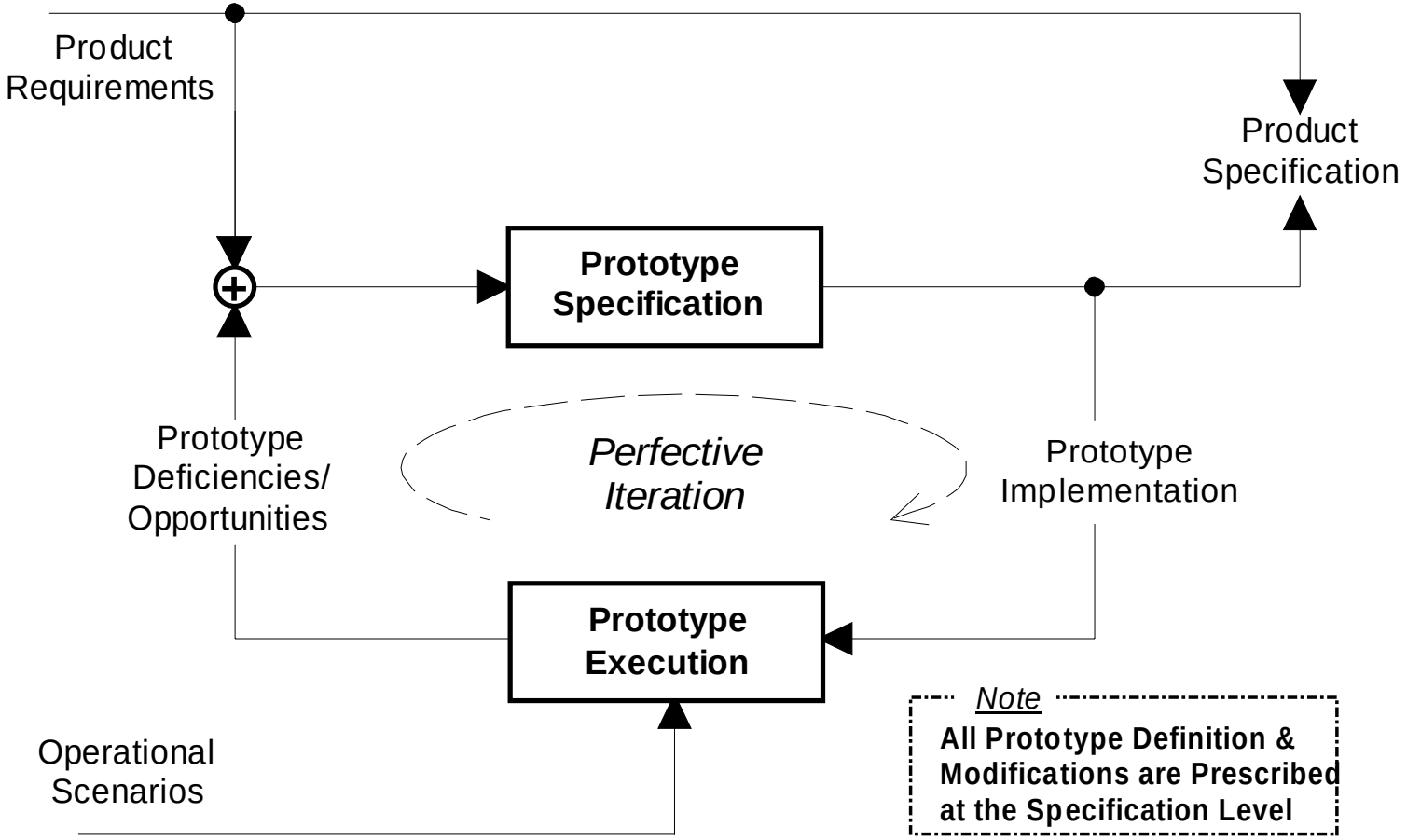
.	ANALYSIS	SIMULATION
SCOPE	General Conclusions	Particular Conclusions
ORIENTATION	Equivalence Classes (Breadth)	Problematic Scenarios (Depth)
DOMAIN	Encompassing Properties	Selective Subset of Behaviors
KEY	Tractable yet Admissable Model Simplifications	Representative Scenario Selections
MECHANISM	Reasoning/Consequences	Stimulation/Observations
MODE	Static/Detached	Dynamic/Tangible
CLOSURE	Deductive	Inductive

Prototyping Approaches
CRITIQUE

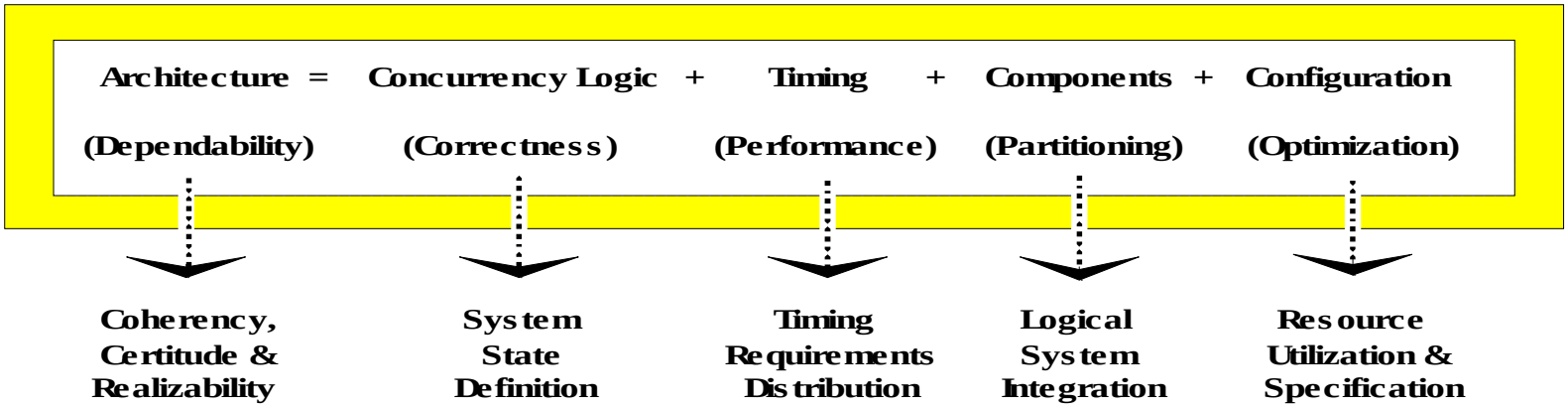
- **THROWAWAY**
  - Difficulty in recovering semantics from prototype
  - Lack of discipline & focus
- **EVOLUTIONARY**
  - Tendency to diverge
  - Inclination to poor structure
  - Lack of discipline & focus
- **SPECIFICATION-DRIVEN**
  - Recovery of semantics unnecessary (specified before prototype implementation)
  - Specification evolves, not the prototype
  - Enforcement of discipline & focus

Specification-Driven PROTOTYPING



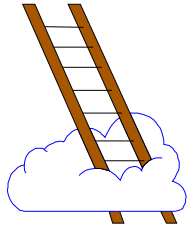


Four-Stage Prototyping PROGRESSION



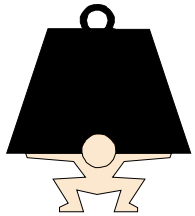
- Same Architecture Model(s) Evolved over the 4 Stages
- Outputs of Logic & Quantitative Parameters for Specifications

Specification Language: *HIGHER-LEVEL* STATECHARTS (HLSs)



- HLSs are Arbitrarily *Scalable* Communicating Extended Finite-State Machines (CEFSMs)
- Statecharts Subgraphs are Process Types
- Tokens within them are Instances of Data Types
- Transition Rules Syntax is Augmented 1st-Order Logic
  - Scalability
  - Absolute timing
  - Stochastic effects
- Multi-Level State Definition

Higher-Level Statechart
EXPRESSIVENESS



Property	Realization	Role
Communicating	External events	Message passing Notification Request Timeout
Extended	Tokens	Local state data Message parameters
	Timing	Process duration Scheduling times Transmission delays
	Stochastics	Timing variation Demand variability Stochastic decisions
Finite-State Machine	Statechart subgraph	Active objects

Note: Higher-level statecharts are also ***scalable*** with no changes in representation.

Summary & Conclusions

- **System-Level Infrastructure is *the* Dominant Influence on Most Aspects of Dependability**
  - Confront explicitly during system architecting
- **Prototype Execution Identifies, Informs & Justifies Architecting Commitments**
  - Stimulates & amplifies reasoning processes
- **Specification-Driven Prototyping is Vital to Imparting Discipline & Exactitude**
  - Overcomes problems of customary approaches
- **Prototyping can Improve Product Specifications**
  - "*Globally o*ptimized" nominal values & tolerances
  - Global concurrency logic

References

1. Leveson, N.G.: Safeware - System Safety and Computers, Addison-Wesley, 1995.
2. Perrow, C.: Normal Accidents, Princeton University Press, 1999.
3. Atkinson, C., T. Moreton, & A. Natali: Ada for Distributed Systems, Cambridge University Press, 1988.