A Case Study in Predictive Program Management

John Welsh, Bipin Chadha, Zach Horiatis

Lockheed Martin
Advanced Technology Laboratories

Oct. 2004
Problem

- Major programs often encounter major cost and schedule problems despite detailed and comprehensive plans
  
  “Despite their superiority, weapon systems routinely take much longer to field, cost more to buy, and require more support than investment plans provide for.”

  Defense Acquisitions: Assessments of Major Weapon Programs, GAO Report, GAO-03-476, May 2003

- Complex interrelationships cause control of program cost and schedule to be difficult
  
  ◆ Program managers need a small set of key levers (vs. thousands of tasks, schedules, budgets, risks, action items, …)

  “U.S. Navy and Air Force plans to stiffen penalties for cost overruns”

  Bloomberg – 2/5/2004
What If You Could ...

- Know that a software project would be delayed before it was reported
- See management data in real-time and play “what-ifs”
- Recognize cross IPT shortfalls and impacts immediately
- See third and fourth level effects of change rather than those immediately apparent
- Do multi-variant cost sensitivity analysis on the fly

Recognize complex impacts of change at the speed of information rather than speed of “status report” delivery
Levels of Understanding

- **Events (Reactive)**
  - This event happened therefore we must take this action
  - Q3 earnings results announced leading to drop in stock price

- **Patterns of behavior (Responsive)**
  - Interpretation of current events in the context of long-term historical change
  - Understand a pattern of behavior
  - Product lifecycles follow the “S” curve

- **Systemic structure (Generative)**
  - Addresses the question - what causes the pattern of behavior
  - Attempt to address the underlying causes of behavior

-- Senge, P., The Leader’s New Work: Building Learning Organizations, SMR, Fall, 90
Prediction on large complex programs is feasible because …

Most program “surprises” result from a small number of key elements

- Dependencies
- Systemic behavior patterns

These are normally not considered in planning

- Hence root causes of program surprise go undetected

Information for effective analysis is straightforward / easy to obtain

Analysis methodology tolerates parameter variation and imprecision
Comprehensive Program Management

1st Generation Program Management
- IPT’s assessments: Predictive but subjective
- Provides drilldown capability

2nd Integrated Management Framework
- Linked data elements
- Configuration management
- Workflow
- Access control
- Provides views across disciplines, supports change management and traceability

3rd Generation Predictive Program Management
- System dynamic model for program management
- Causal relationships are modeled
- Agent assessments: Predictive and objective
- Provides “look ahead capability” and “what if” levers

MCES 2004 October 26 – 28, 2004
Bipin Chadha
Predictive Program Management Environment

**Prediction Models**

- Statistical Models
- Design Structure Matrices
- System Dynamics Models

- Management Tools (EVMS, Schedule, etc.)
- Collaborative Program Management Environment
- Intelligent Mobile Agents
- Internet Backbone

**Coupling and Dynamics**

**Information Quality**
Key Coupling Factors

Design Structure Matrix

Program Element

Dependency Type

Coupling / Interdependencies
• Program Elements
  — Activities
  — Systems
  — Organizations
• Dependency Types
  — Information
  — Physical
  — Energy
  — Resource

Common dependency patterns inherent across programs enable a small number of models to represent the dominant behaviors.
DSM Dependency Analysis

- For each dependency type, matrices are analyzed to find task or system groupings that are most important to watch
- Supports automated analysis
- Easy to interpret, visualize, and analyze
  - Rows and columns: Program elements
  - Matrix entries: Coupling strengths
  - Principle eigenvalues: Key work / design modes; determine the “rate and nature of convergence” in the presence of change
  - Principle eigenvector: Ranks tasks/subsystems by their “relative impact” on overall schedule / cost

Dependency analysis highlights the subsets programs which require closer attention / monitoring, or need to be decoupled.
DSM Analysis

- Design mode: Group of tasks that are closely related and cause rework. Characterized by eigenvalues and corresponding eigenvectors.
- Eigenvalues and Eigenvectors determine the rate and nature of convergence of process.
- Magnitude of eigenvalues identifies the geometric rate of convergence:
  - Large positive eigenvalues have greater impact.
  - Ignore complex eigenvalues with negative real component.
  - Negative and complex eigenvalues describe damped oscillations.
- Eigenvectors characterize relative contribution of tasks to rework.
Selected Design Modes can be Analyzed for Systemic Behavior Patterns

Interdependencies / Coupling

- Program Elements
  - Activities
  - Systems
  - Organizations

- Dependency Types
  - Informational
  - Physical
  - Energy
  - Resource

Dynamic Patterns

- Balancing Loop
- Reinforcing Loop
- Hidden Rework
- Fixes that Backfire
- Limits to Growth
- Shifting the Burden
- Tragedy of Commons
- Accidental Adversaries
- Success to the Successful
- Escalation
- Drifting Goals
- The Attractiveness Principle
- Growth and Underinvestment (core patterns)

Common systemic patterns inherent across programs enable a small number of models to represent most of the dominant behaviors (80% solution).
Systemic Behavior

- Behavior of a system that emerges due to dynamic interactions among components

- System characteristics
  - System structure drives system behavior (especially the feedback loops)
  - System exhibits properties and behaviors different from the parts
  - Everything is connected to everything
  - There are no independent variables
  - Many effects are indirect and delayed
  - Many dependencies are non-linear
  - Behavior is dynamic

Systemic Behavior Can be a Significant Source of Risk and Surprise
System Dynamics Modeling (SDM)

- **Observation**
  - Problem arises over and over again
  - You apply the fix, there is some relief, but the problem surfaces again

- **What is going on**
  - You are applying a symptomatic solution that alleviates symptoms immediately but exacerbates the problem over time
  - The problem causes symptoms to reappear after some delay
  - Since the fundamental solution has a delay built into it (before you see results) it forces you to take the symptomatic solution in the short term, thus trapping you in the vicious cycle

"Shifting the Burden" Pattern

Source:
The Fifth Discipline Fieldbook by Senge, et al
Where Have These Techniques Been applied Before?

- **DSM**
  - Developed at MIT
  - Applied in Automotive industry
  - Applied on some DoD programs
  - MIT runs regular courses on it for program managers
  - Significant body of literature exists that supports this technique

- **SDM**
  - Developed at MIT
  - Applied in a wide range of industries to analyze systemic problems
  - Applied on some DoD programs
  - Significant body of literature exists that supports this technique
  - Several patterns have been well documented

Individually both techniques have been proven useful, but haven’t been tied into a comprehensive solution.
Intelligent Mobile Agents for PM

Agent Tasks

- Information gathering
- Data analysis resulting in R/Y/G labeling
- Determine each WBS’s associated R/Y/G cost / schedule / risk / etc.
- Trend analysis
- Historical data analysis
- Action Item monitoring
- Critical path monitoring
- Generate alerts
- Earned value analysis
- Predictive analysis
- Recommendations

Leveraging Lockheed Martin agent technology to manage a SDM; user sees

- Information
- Not agents
- Not volumes of data
Case Study: DSM Based Rework Analysis

- DSM analysis shows one dominant design mode (eigenvalue) and relative impact of tasks (eigenvector) on the overall schedule.
- Analysis is robust in the presence of parameter variation / uncertainty.
- Analysis predicts approximate percent rework for the tasks due to dependencies / coupling.
  - Program manager can use this analysis to verify estimated rework.
  - Analysis can be used to evaluate the impact of task restructuring on expected rework (which is often the cause of surprise towards the end).
Case Study: DSM used to Create System Dynamics Model (SDM)

DSM can identify a small subset of dependencies that require detailed modeling via SDM.
Model is successfully predicting the observed data.
Conclusions

- **Systemic model demonstrates the behavior in the presence of rework and feedback**
- **The results of the model are broadly applicable**
  - All iterative processes will demonstrate this behavior to a certain extent (rework pattern)
- **Model is successfully predicting the crossover effect**
  - This is where the status goes from green to red overnight
  - Early analysis can predict this effect and program managers can plan accordingly to prevent surprise
  - Model can be run anytime with latest data to detect possibility of crossovers (ideal task for an agent)
- **Planning to deploy capability on programs**