

A Framework for Estimating the Cost of Fixing the Year 2000 Problem

Neal D. Hulkower, Ph.D.

John A. Vitkevich

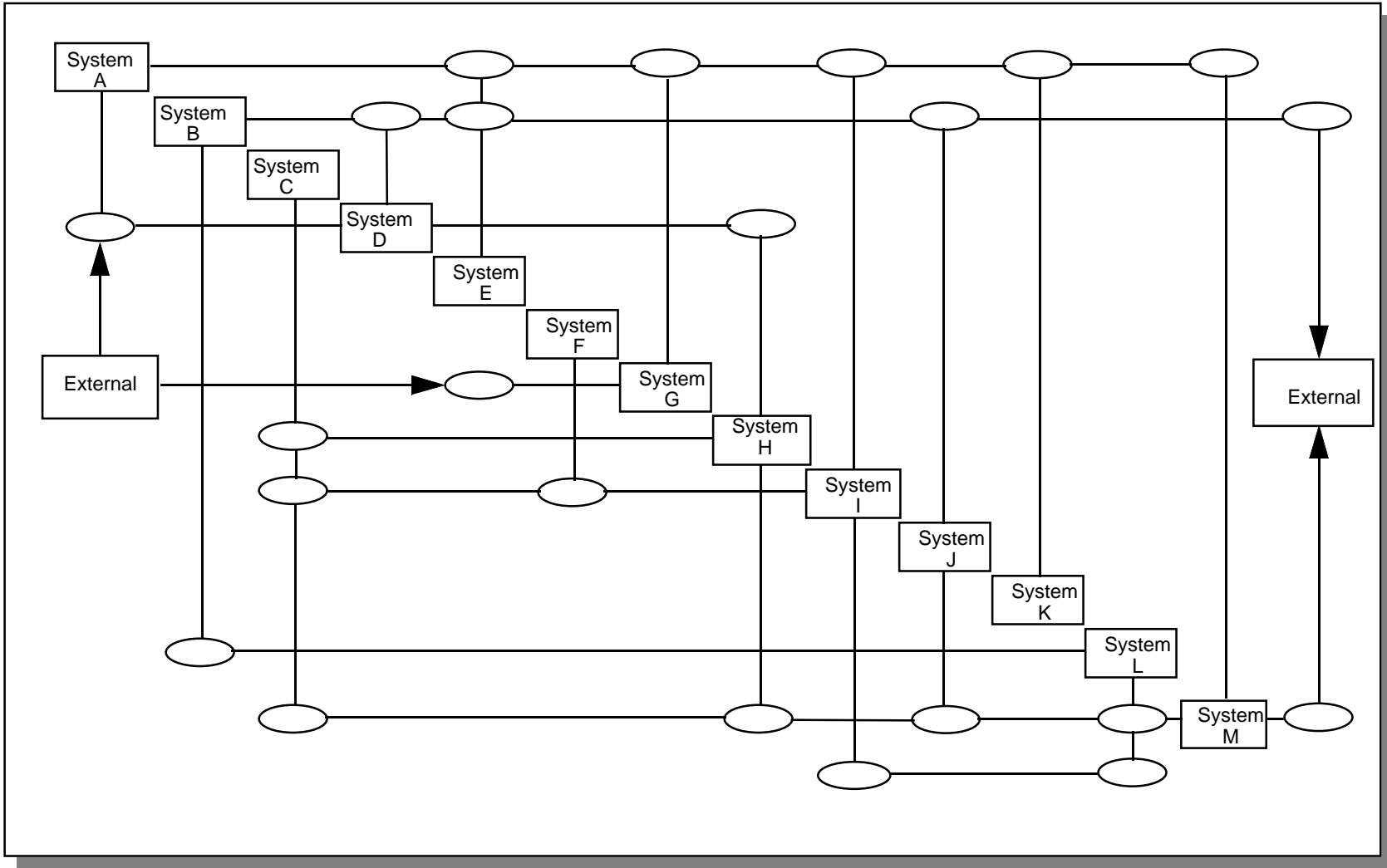
13 February 1997

MITRE

Outline

- **Characterizing the Problem**
- **Steps in Addressing Your Y2K Problem**
- **Determining the Cost**
- **Cost Uncertainty Issues**
- **Conclusions**

Networked System of Systems



Unique Challenges of System-of-Systems Costing

- **Increased complexity**
- **Obtaining agreement on what systems and interfaces to include**
- **Consistency across all systems**
 - **Common set of cost inputs**
 - **Consistent treatment of cost elements**
- **Shared assets and costs between systems**
 - **Must identify to avoid double counting**
- **Intersystem network testing**
- **Multiple acquisition approaches**

Five Phase Problem Resolution Cycle

- **Awareness**
 - **Motivating and mobilizing**
- **Assessment**
 - **Code scans**
 - **Test planning**
 - **Detailed equipment inventories**
 - **Detailed cost estimates**
- **Renovation**
 - **Code fixes**
 - **Equipment upgrades and COTS replacement**
 - **Individual system testing**
 - **Set up intersystem network test assets**

Five Phase Problem Resolution Cycle (Concluded)

- **Validation**
 - Intersystem network testing
 - Critical function testing
- **Implementation**
 - Test fixes and rework
 - Deployment and cutover preparation
 - Rollover operations

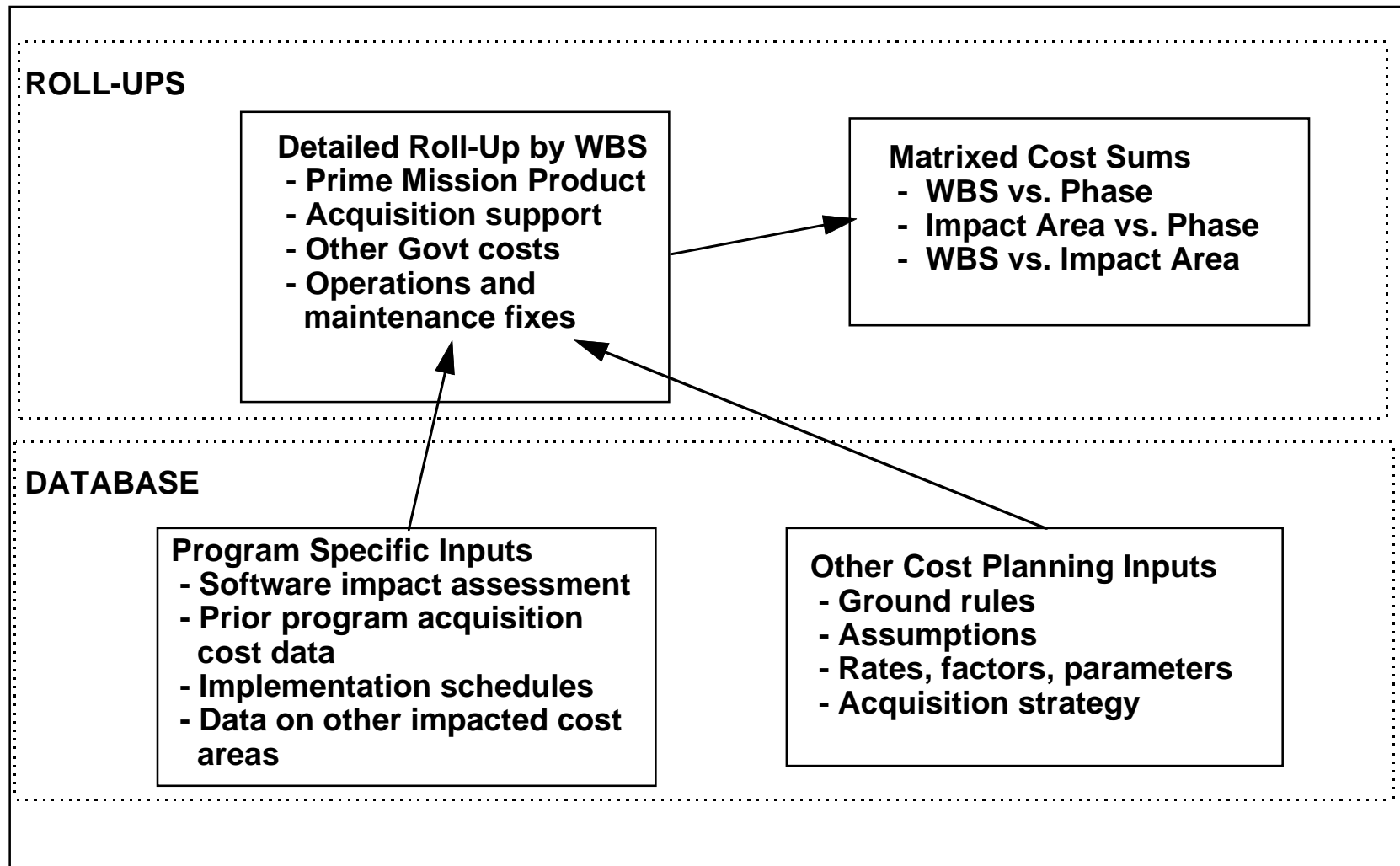
Steps in Addressing Your Y2K Problem

- **Identify the major system nodes, including legacy hardware and software operating beyond 2000**
 - **Each node consists of hardware and software required to perform a particular mission or set of missions**
 - **In the Air Force, these are frequently single acquisition programs**
- **Identify node to node interfaces**
- **Identify external interfaces**
- **Identify system security impact**
- **Determine Y2K impacts on each node**

Steps in Addressing Your Y2K Problem (Concluded)

- **Schedule fixes as part of normal maintenance**
- **Identify and fix code not normally maintained**
- **Work with vendors to address Y2K fixes to COTS packages**
- **Perform validation testing**
- **Deploy and have contingency plans for critical missions**
- **Prepare for the rollover day and its consequences**

Cost Estimating Framework



Cost Areas Impacted

- Applications software
- System security
- Asset augmentation (e.g., firmware, interfaces, chip redesign, additional data storage devices)
- Intersystem network test
- Critical function test
- Y2K tools and environment
- COTS/GOTS equipment
- Acquisition support for operations and maintenance release
- System downtime and associated workarounds
- Other government costs
- Deferral of previously planned cost saving measures
- Contingency reserve

Example of a Cost Work Breakdown Structure by Phase

<u>Prime Mission Product</u>	<u>Awareness</u>	<u>Assessment</u>	<u>Renovation</u>	<u>Validation</u>	<u>Implementation</u>
Nonrecurring Engineering			√		√
Hardware			√		√
COTS Software			√		√
Software Development			√		√
Integration and Assembly			√		√
<u>Acquisition Support</u>					
Program Management			√		√
Systems Engineering*		√	√	√	√
Test and Evaluation*		√	√	√	√
Data*			√		√
Site Activation			√		√
Peculiar Support Equipment			√		
Spares			√		
Training		√	√		√
Vendor Support		√	√	√	√
<u>Other Government Costs</u>					
Engineering Services*	√	√	√	√	√
Technical Studies*	√	√	√	√	√
Government Furnished Equip.		√	√	√	√
Test Facilities and Support*			√	√	√
Government Travel	√	√	√	√	√
* Includes system security engineering					

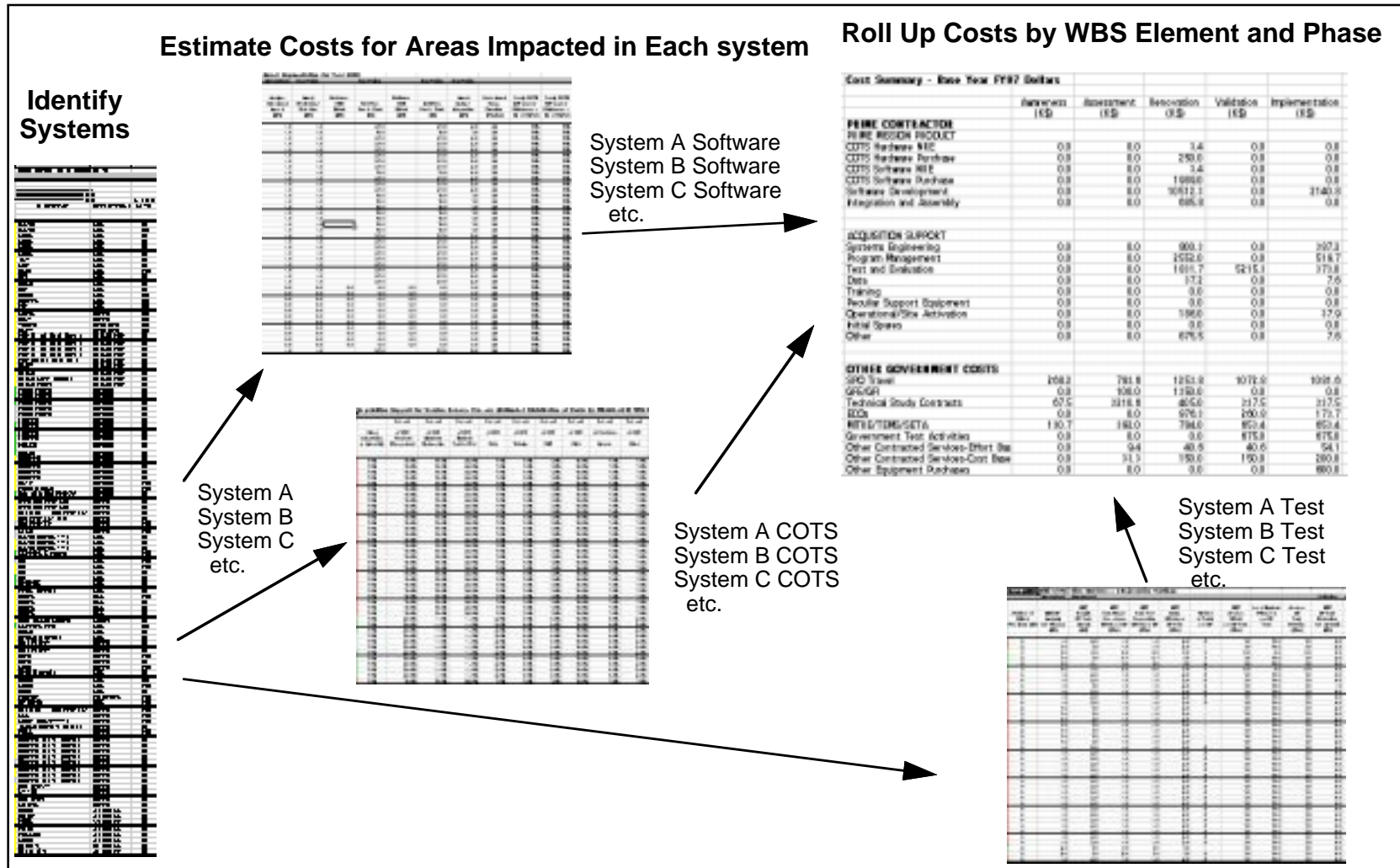
Determining the Cost

- **Standard cost estimating methodologies for possible use**
 - **Cost estimating relationships (CERs):**
e.g., $Effort = A(KSLOC)^B$ where effort is in staff months and KSLOC is thousands of source lines of code being developed
 - **Parametric models**
 - **Analogy based on historical data, e.g., productivity factors**
 - **Engineering judgment, e.g., direct estimate of staffing for program level activities**
 - **Vendor quotes, e.g., for COTS hardware and software**
- **Selection of methodology depends on what cost is being estimated and what type of data are available**
- **Expect to use a mix of methodologies**

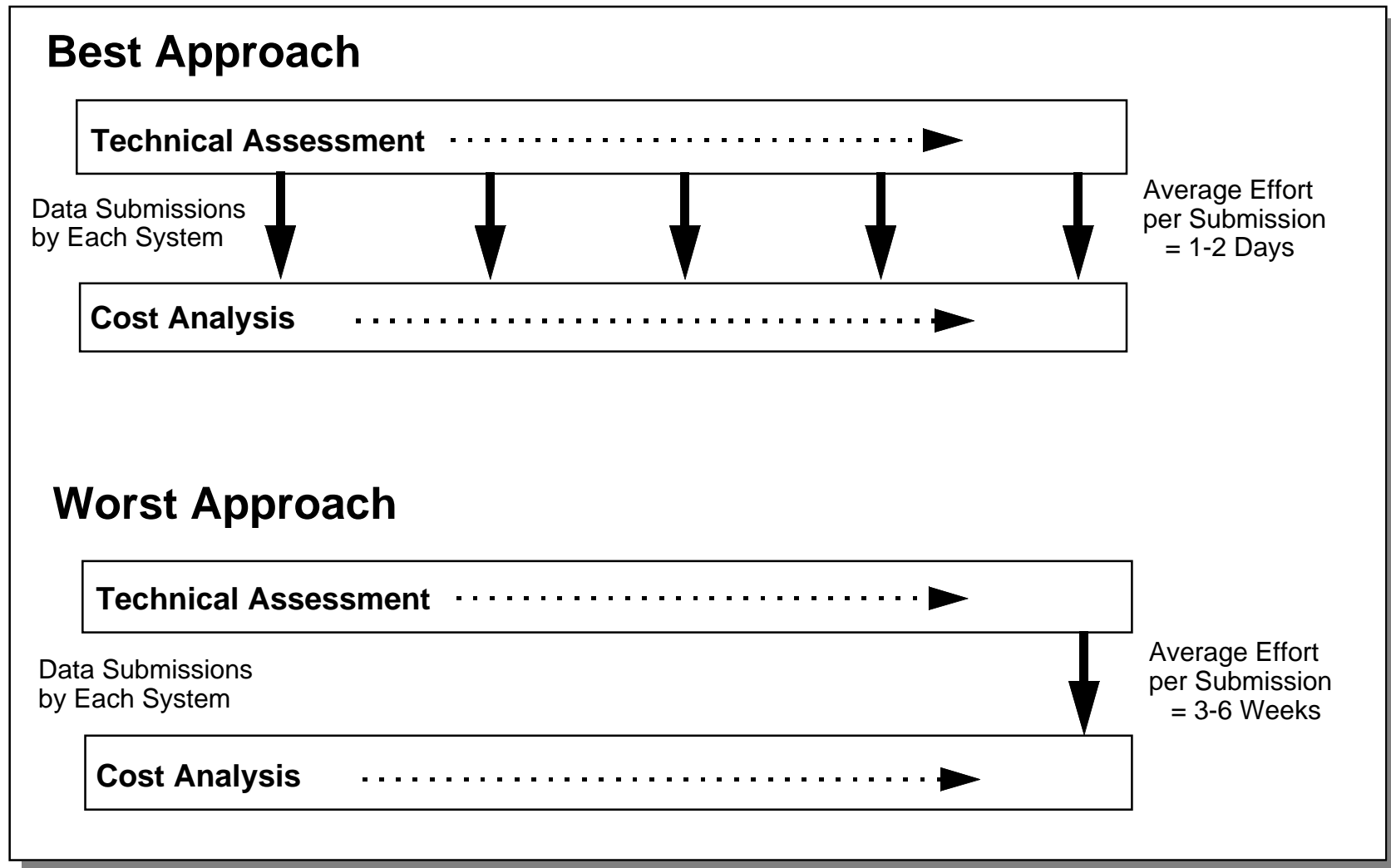
Estimating Software Maintenance Costs

- **If software is being maintained, use current productivity but know what is included in your measures**
- **Most of the parametric models claim to estimate maintenance costs**
 - **Additional cost driver is “annual change traffic” (percent of code that requires re-development annually)**
 - **Useful for planning when systems are in development**
- **Most models fail to fully address maintenance issues**
 - **No emphasis on effort to understand architecture or code functionality**
 - **Re-test and integrate effort is likely larger than that for “touched” code**

Cost Analysis Technique



Cost Assessment Process



Cost Uncertainty Issues

- **Total amount of software and amount effected**
- **Security impacts**
- **Unmaintained code from legacy systems**
- **COTS hardware and software**
- **Firmware**
- **Testing**
- **Number of systems to be included**
- **Quality of data submitted by individual systems**
- **Appearance of new cost areas and impacts**
- **Backup and workaround measures, and their implementation**
- **Contingency reserve for emergencies and rebuilding**

Conclusions

- **For complex systems, achieving Y2K compliance is a difficult engineering and management challenge**
 - **A detailed understanding of the hardware and software components of each system and knowledge of system interfaces are essential but likely to be incomplete**
 - **The “devil is in the details” so build a work breakdown structure to an appropriate level of cost visibility**
 - **Legacy systems abound**
 - **Knowledge of how COTS vendors are making their products Y2K compliant is necessary**
 - **Interactions between systems must be tested**

Conclusions (Concluded)

- **Expect existing commercially available software cost estimating models to be of little help**
- **Cost to your program could be:**
 - **Negligible, if system is standalone with no interfaces or interoperations, and code is well maintained**
 - **Large, if systems are interconnected with many interfaces passing high volumes of data, and code is not maintained**
 - **Somewhere in between**
- **Year 2000 survival won't happen by itself**