Use Cases for Evaluation of Machine-Based Situation Awareness

Kenneth Baclawski
Anna Chystiakova
Kenny C. Gross
Dieter Gawlick
Adel Ghoneimy
Zhen Hua Liu

Situation Awareness

- Situation awareness (SA) is the perception of environmental elements and events with respect to time or space, the comprehension of their meaning, and the projection into the future. (Endsley 1995).
- SA is important for successful decision making
- Decision making is necessary for humans, machines and collaborations among both.
- However, SA for humans differs from SA for machines.
- Evaluating SA will also differ.

Human vs Machine Decision Making

- Machine
  - More information
  - More complex models
  - More rapid response time
  - Multiple levels of decision making
  - Deterministic and explainable
  - Cost-benefit model is appropriate for evaluating SA
- Humans
  - More flexible in response to novel situations
  - Ultimately, humans must still play a role
  - Nondeterministic and not necessarily explainable
  - Level of accuracy is used for evaluating SA

Human Situation Awareness

- Endsley Model
  - Very useful model of human decision making
  - Emphasis on improving accuracy of SA
  - Lacks detailed process and data models
- Cognitive Architectures
  - Examples are ACT-R and Soar
  - Provide process and data models
  - Purpose is to refine theories of cognition
  - Limited concern with performance or efficiency

Machine-Based Situation Awareness

- KIDS Model
  - Detailed model of processes and data models
  - Emphasis on performance and efficiency
  - Not concerned with theories of cognition
- Combined Models
  - KIDS is compatible with the Endsley model
  - Useful for Human-Machine collaboration
  - The best model or combination of models depends on the use case

Evaluation Use Cases

- The most appropriate model for SA and its evaluation depend on the use case
- We present three examples of use cases:
  - No Trouble Found Use Case
  - The Bullwhip Effect Use Case
  - Cloud Services Use Case
- Each is described and an evaluation technique is developed.

No Trouble Found Use Case

No Trouble Found (NTF)

- Components can be returned to the supplier under contract provisions
  - Returned due to an alarm
  - But 25% to 70% function correctly!
- Estimated cost of this problem is $2B/year
- An example of the loss of SA
  - The perception of the component status is incorrect
  - Results in an incorrect decision

Causes for NTF

- Primary causes of NTF
  - Transient/Intermittent Faults
  - Threshold Limits on Noisy Physical Variables
  - Sensor Degradation Events
  - Errors During Testing and Diagnosis
- Testing can help but
  - There is a cost for testing
  - Tests are not always definitive

Proposed Decision Making Process

- Goal is to maximize the net benefit of testing
- Tests are performed in the optimal order
  - The problem is to determine the optimal order
  - Assumes that tests are statistically independent
- The evaluation of the net benefit is presented on the next slides

Mathematical Model Part 1

- Tests $T_1, T_2, \ldots, T_n$
- Costs $C_1 < C_2 < \ldots < C_n$
- Probabilities of outcomes of a test *T*
  - p = Prob(Component is defective)
  - q = Prob(Component is working properly)
  - r = Prob(Unable to determine)
  - $p + q + r = 1$
- Component value is *V*

Mathematical Model Part 2

- The net average benefit of T_i is $f(i) = q_i V - C_i$
- If the tests are performed in the order $\{i_1, i_2, \ldots, i_n\}$
- Then the total net average benefit is:

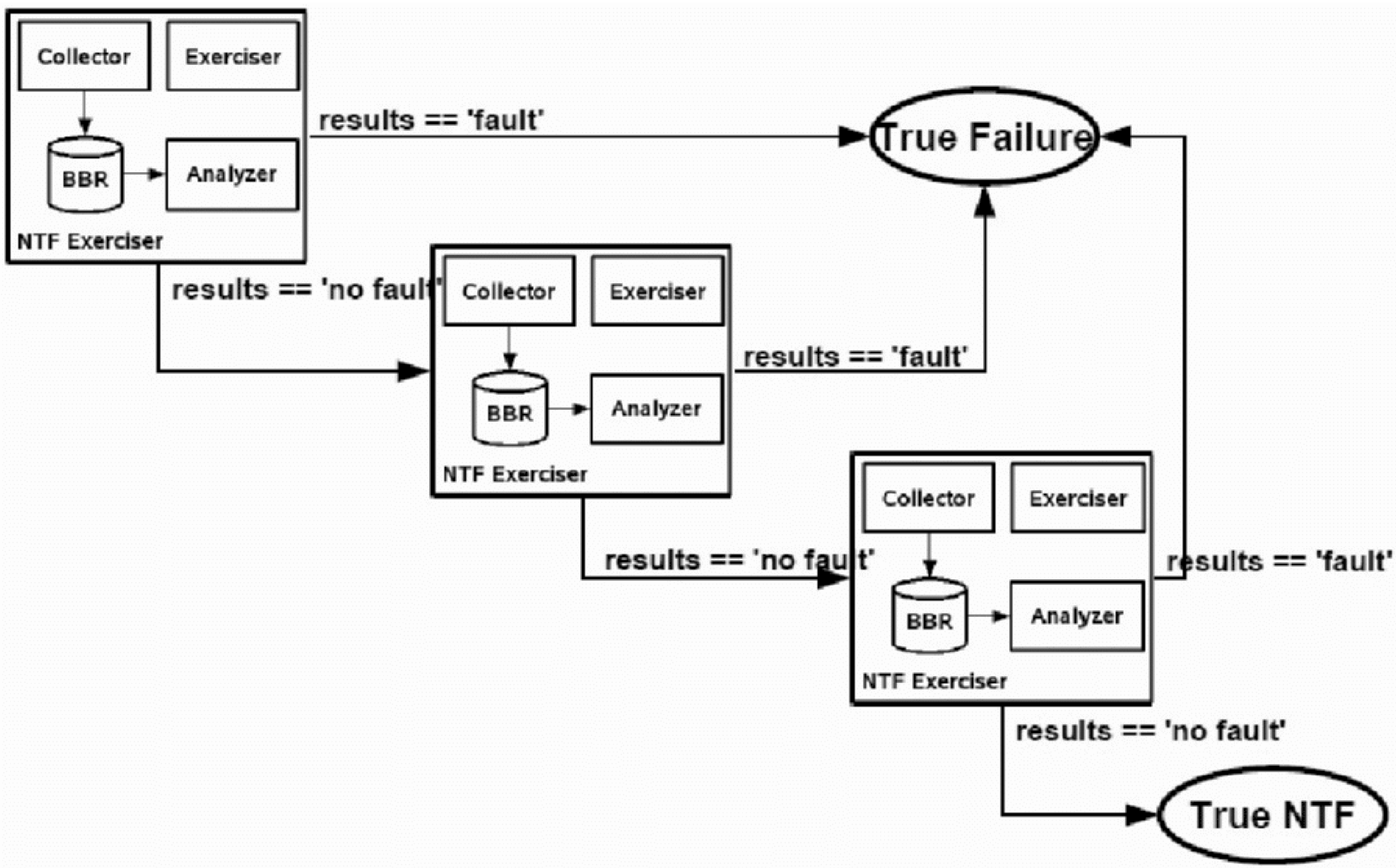
$$f(i_1, i_2, \ldots, i_n) = \sum_{j=1}^{n} \left(\prod_{k=1}^{j-1} r_{i_k}\right) f(i_j)$$

- The optimum order is the one that maximizes the formula above.

Variations

- The same test can be repeated
  - Useful only if results are statistically independent
  - Complicates analysis but still feasible
- Statistical dependencies
  - Analysis is still possible but much more complicated
- Machine learning (ML)
  - Potential approach for improving SA

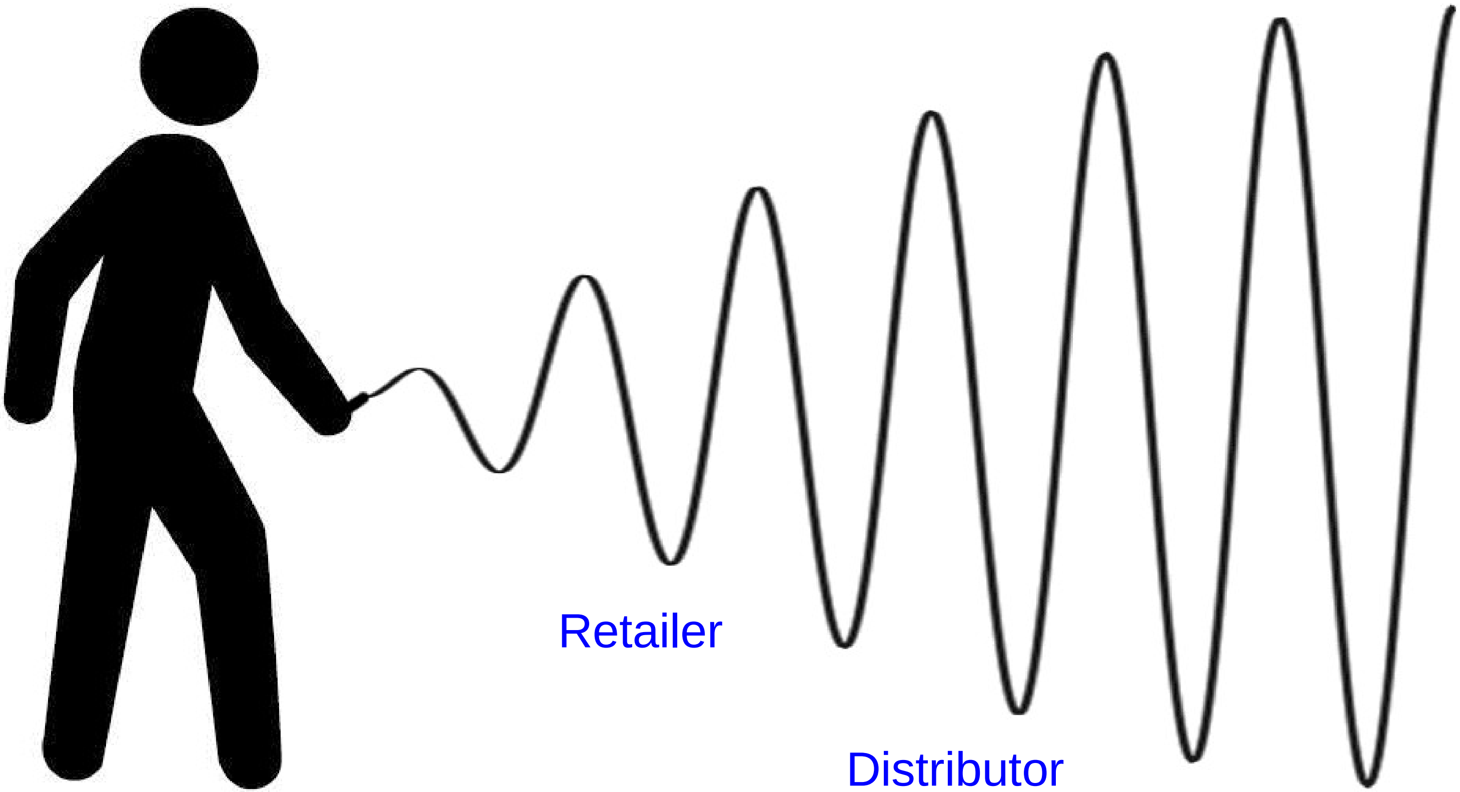
Example of Test Sequence with 3 Tests



Bullwhip Effect Use Case

Bullwhip Effect

- Also called the Forrester Effect
- Business forecasts overcompensate in response to shifts in demand
  - Results in increasing swings in supply
  - Much more serious for multiple links in a supply chain
- Even when people have perfect information optimum performance is difficult to achieve
- Also commonly observed in software systems

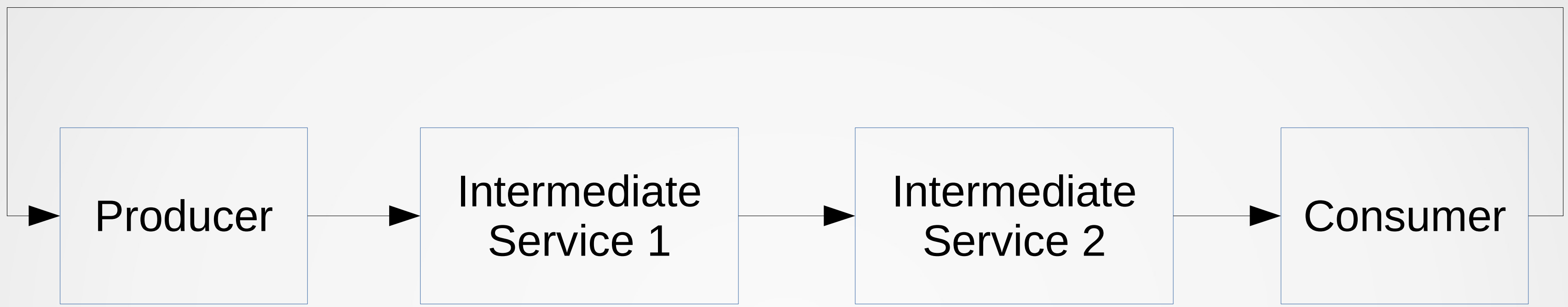


Source: https://en.wikipedia.org/wiki/Bullwhip_effect

Approaches

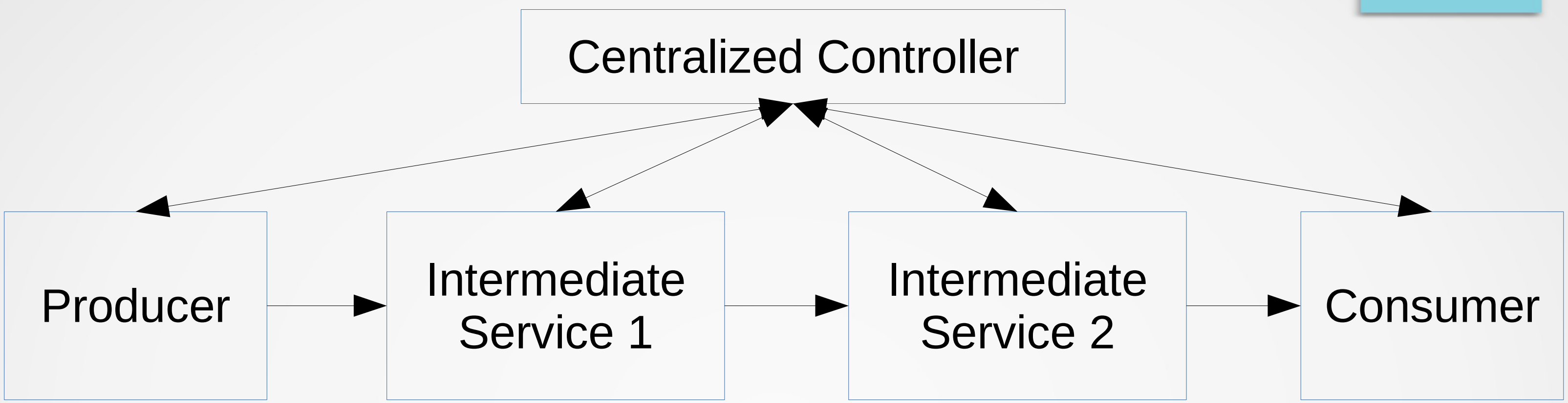
- Ignore the problem
- Centralized decision making
- Feedback control techniques
- Self-Controlling Software Model

Chain of Services



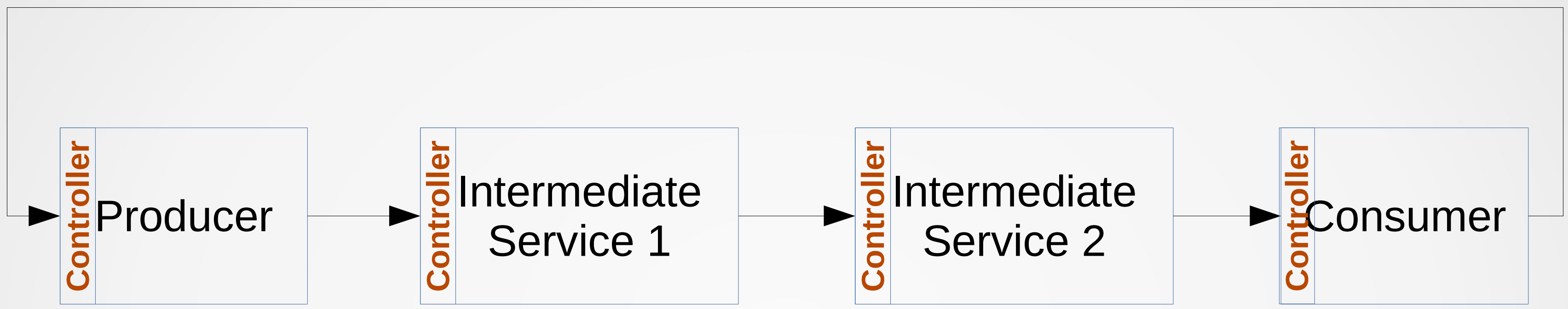
- Advantages
  - Easy to program
  - Autonomous components: modular, resuable, etc.
- Disadvantages
  - Prone to instability (e.g., bullwhip effects)
  - Far from optimal performance

Centralized Control of Chain of Services



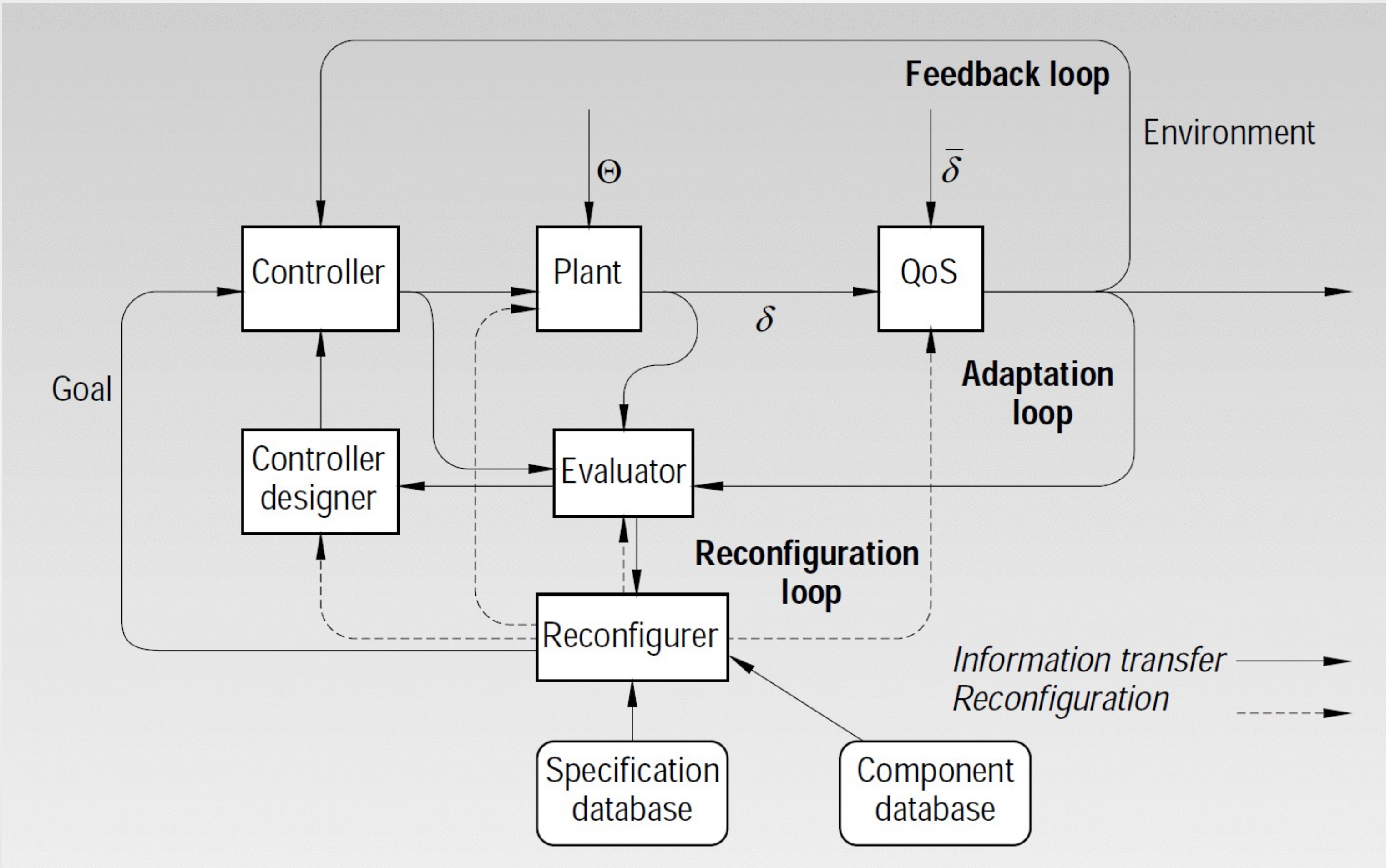
- Advantages
  - Can come close to eliminating instability
- Disadvantages
  - Substantial reprogramming is required
  - Loss of modularity, reuse, etc.
  - Instability may still occur due to communication delays

Controlled Chain of Services



- Advantages
  - Easy to program
  - Autonomous components: modular, resuable, etc.
  - Reduces likelihood of instability
- Disadvantages
  - Not optimal, but still good performance

Self-Controlling Software Model



Feedback Control Theory

- Time models
  - Discrete time: z-transform
  - Continuous time: Laplace transform
- A component is modeled using the transfer function in the transform space
- We have used discrete time and the z-transform in the paper, but Laplace transform techniques are similar.
- Feedback control theory is highly developed in Engineering
- Econometric models also use feedback control theory
- Computer science seldom uses feedback control theory

PI Controller

- Transfer function

$$(K_P + \frac{K_I z}{z-1})(\frac{1}{z-1})$$

- K_P is the coefficient of proportional control
- K_I is the coefficient of integral control
- Not used: K_D the coefficient of derivative control

Modeling a Service Component

- Software is complicated!
- However, performance can be bounded as in computational complexity theory
- Worst case analysis
  - Component transfer function is a constant G
  - Probability of exceeding the worst case bound can be determined empirically

Modeling Controlled Chain of Services

- Transfer function of the chain of services:

$$F(z) = \prod_{i=1}^{n} (K_P^{(i)} + \frac{K_I^{(i)} z}{z-1})(\frac{1}{z-1}) G_i$$

- Transfer function of the closed loop:

$$H(z) = \frac{F(z)}{1+F(z)} = \frac{R(z)}{(z-1)^{2n} + R(z)}$$

- where

$$R(z) = \prod_{i=1}^{n} ((K_P^{(i)} + K_I^{(i)}) z - K_P^{(i)}) G_i$$

Preventing Instability

- Closed loop stability criterion
  - Every pole of $H(z)$ is inside the unit circle
  - If ζ is a pole of $H(z)$, then norm(ζ) < 1.
- The PI coefficients must be tuned to keep the poles away from the unit circle
  - Even being close to the unit circle is problematic
  - Stability generally requires throttling the services
- One can now perform a cost-benefit analysis
  - The cost is due to the throttling of the services
  - The benefit is increased probability of stability

Cloud Services Use Case

Cloud Services

- Increasingly popular service
  - Sharing resources reduces fluctuations
  - Resources are more fully utilized
- Cloud service providers
  - Must satisfy contractual service level agreements (SLA)
  - SLA failures entail financial penalties
- Bullwhip effects are commonly observed

Managing Cloud Services

- Cloud services manage many resources
  - Processor time
  - Memory
  - Network bandwidth
  - Storage
  - Database servers
  - Other servers
- Resource demands affect each other.

Controlling Cloud Services

- Feedback control theory for cloud services requires linear algebra (matrix) methods.
  - Multivariable feedback control
  - Coefficients of the controllers are now matrices
  - The transfer function $T(z)$ is a matrix function of *z*
- Analysis is more complicated
  - If a pole of the determinant $det(T(z))$ is outside the unit circle then the system is unstable
  - Proving stability in general requires diagonalizing $T(z)$

Challenges and Opportunities

- Hierarchical organization of services
- Multiple timescales
- Legal issues
  - Cannot examine internals of client software
  - Requires estimation and modeling techniques
  - Scientific method is well suited to this problem
- Machine learning techniques could be used
- Empirical modeling techniques are also applicable
  - The SCSM includes this as part of the model

Conclusion

Lessons Learned

- NTF Lesson
  - The NTF analysis can be used for evaluating and optimizing SA when a sequence of tests is performed.
- Bullwhip Effect Lesson
  - Control theory techniques can be used to model and understand complex systems.
- Cloud Services Lessons
  - Complex systems require multivariable control theory.
  - Need automated empirical modeling techniques.