Failure Diagnosis in Complex Enterprise Services

Matti Hiltunen

AT&T Labs – Research

IFIP WG 10.4 Meeting

June 2010

Failure Diagnosis: Definition

(a) Detecting that something is wrong (failure detection)

(b) Figuring out what is causing the problem (diagnosis, finger pointing, root cause analysis)

(c) Fixing it (repair).

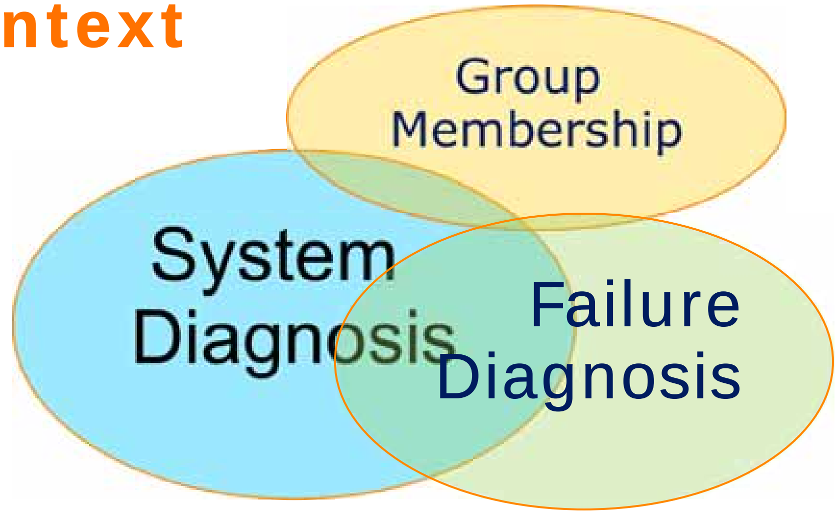
... but

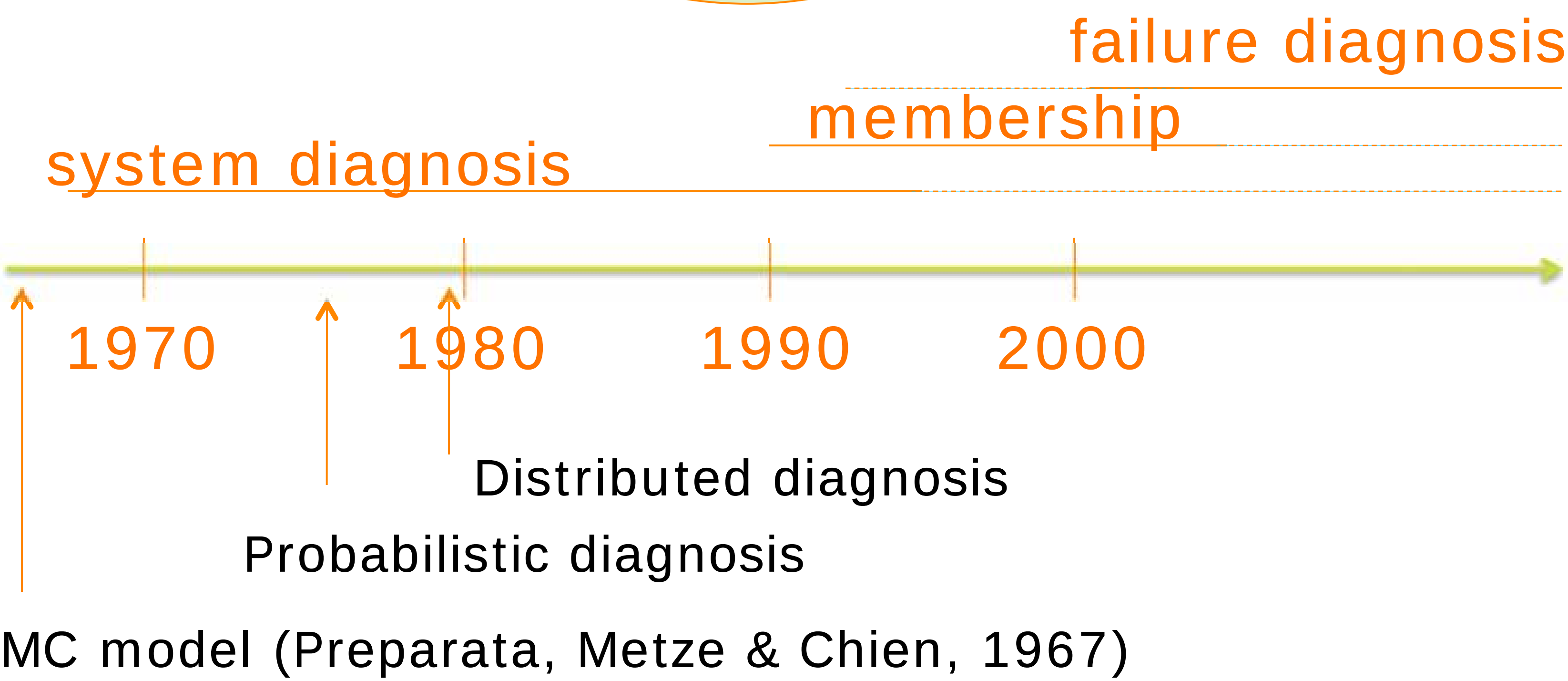
(d) Is it really my problem (is the problem in my system or somewhere else)?

(e) Is it possible for me to fix the problem (e.g., COTS software)?

(f) ...

Context


Group
Membership
System
Diagnosis
Failure
Diagnosis


failure diagnosis
membership
system diagnosis
1970
1980
1990
2000
Distributed diagnosis
Probabilistic diagnosis
PMC model (Preparata, Metze & Chien, 1967)

Personal journey

Distributed Computing => Group membership.

Saw the light!

- Barborak, M., Dahbura, A., and **Malek, M**.. "The consensus problem in fault-tolerant computing." *ACM Computing Surveys,* 25, 2 (Jun. 1993), 171-220.
- M. Hiltunen, "Membership and System Diagnosis", In Proc. SRDS 1995.

Failure diagnosis in enterprise computing (covered in this talk).

Talk outline

1. Challenges of failure diagnosis in enterprise systems
2. Failure Diagnosis in the EMN System
3. Stochastic Model-Driven Diagnosis and Recovery
4. Failure Diagnosis in VoIP Systems
5. Conclusions

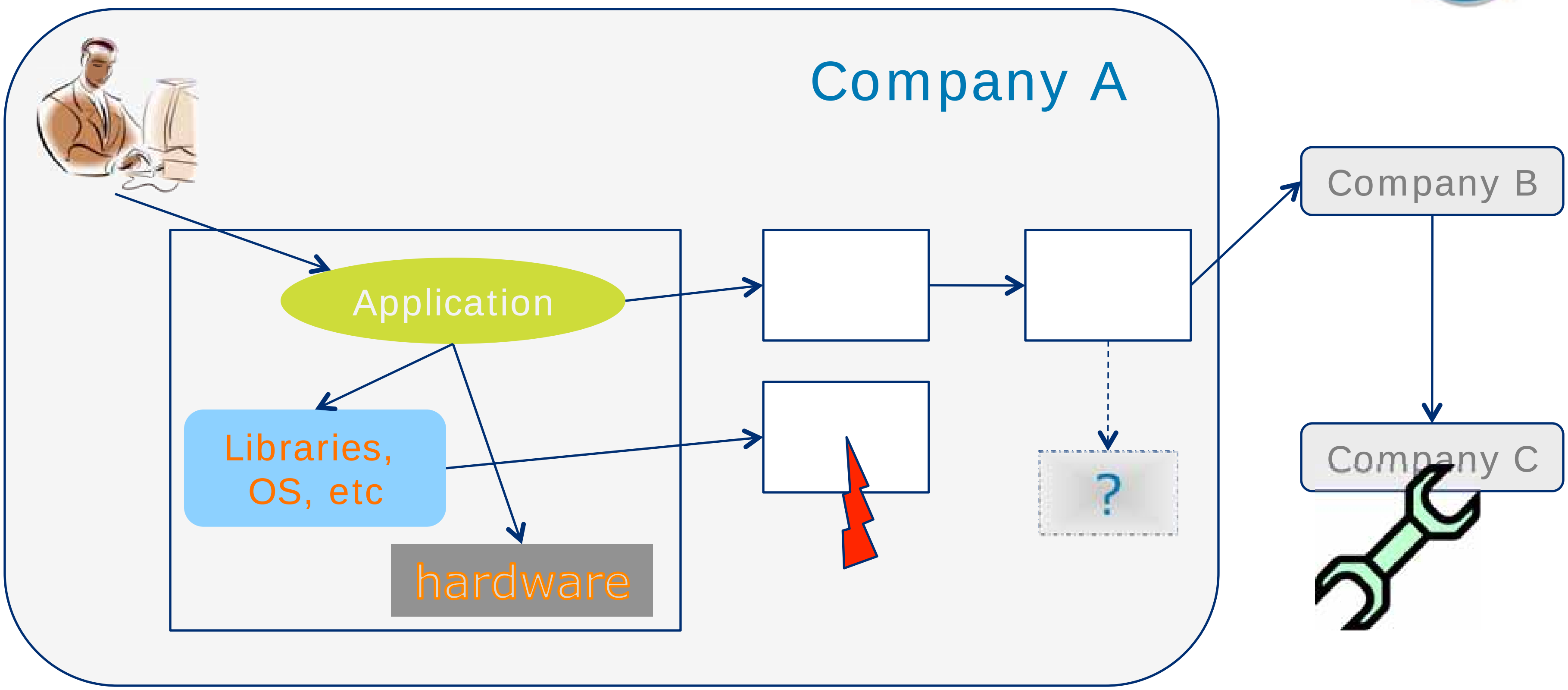
1. Challenges

Failure Diagnosis in Enterprise Systems

Heterogeneity

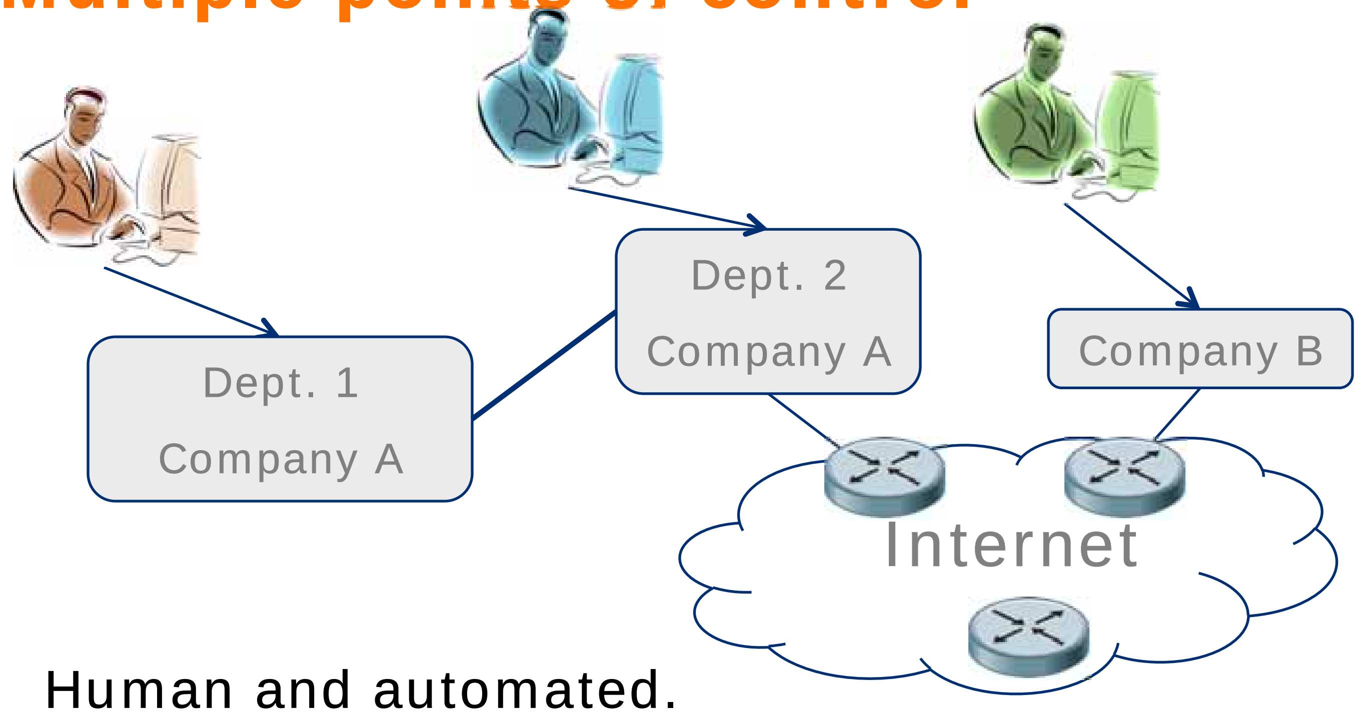
- System consists of heterogeneous components (hardware, OS, applications, networks) with different capabilities w.r.t. diagnosability.
  - Compute and database servers
  - Firewalls, load balancers, etc.
  - Routers, switches, etc
- COTS + home grown
- System often spans multiple types of networks (wired, WiFi, 3G)
- Failures at different layers (network, hardware, software, software configuration).
- Different failure types: crash, performance, omission, quality, value, …
- Permanent and transient failures.
- Chronic failures: low impact, repeating.

Dependencies



- Failures and failure propagation due to (unknown) dependencies between (unknown) elements.
- External (unknown) dependencies.

Multiple points of control



- Human and automated.
  - Unexpected configuration/routing changes.
  - Maintenance outages.
  - Built-in adaptive/reactive behavior.
- In pure diagnosis, you are not a point of control.
- "Debugging by conference call".

Practical challenges

Scale:

- Potentially multiple independent problems may be present in the system at the same time.
- Data volume.
- Number of elements.

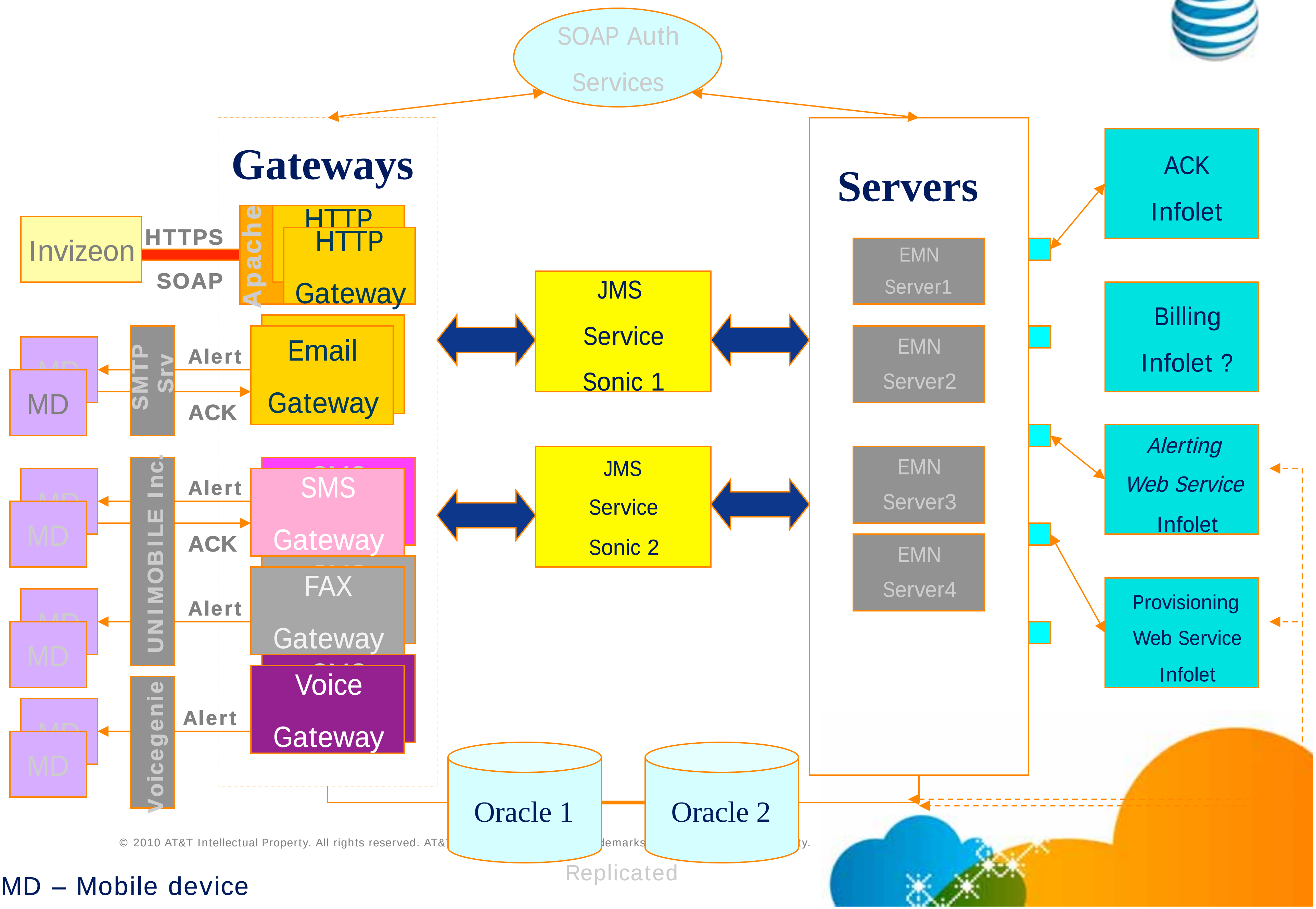
Data quality:

- Limited forms of event data available (e.g., trouble tickets, monitor outputs, some logs, only failure data).
- Data available for only part of the system.
- Gaps in data, delay in data collection.
- Correlation of data from heterogeneous systems.

2. Failure Diagnosis in the EMN System

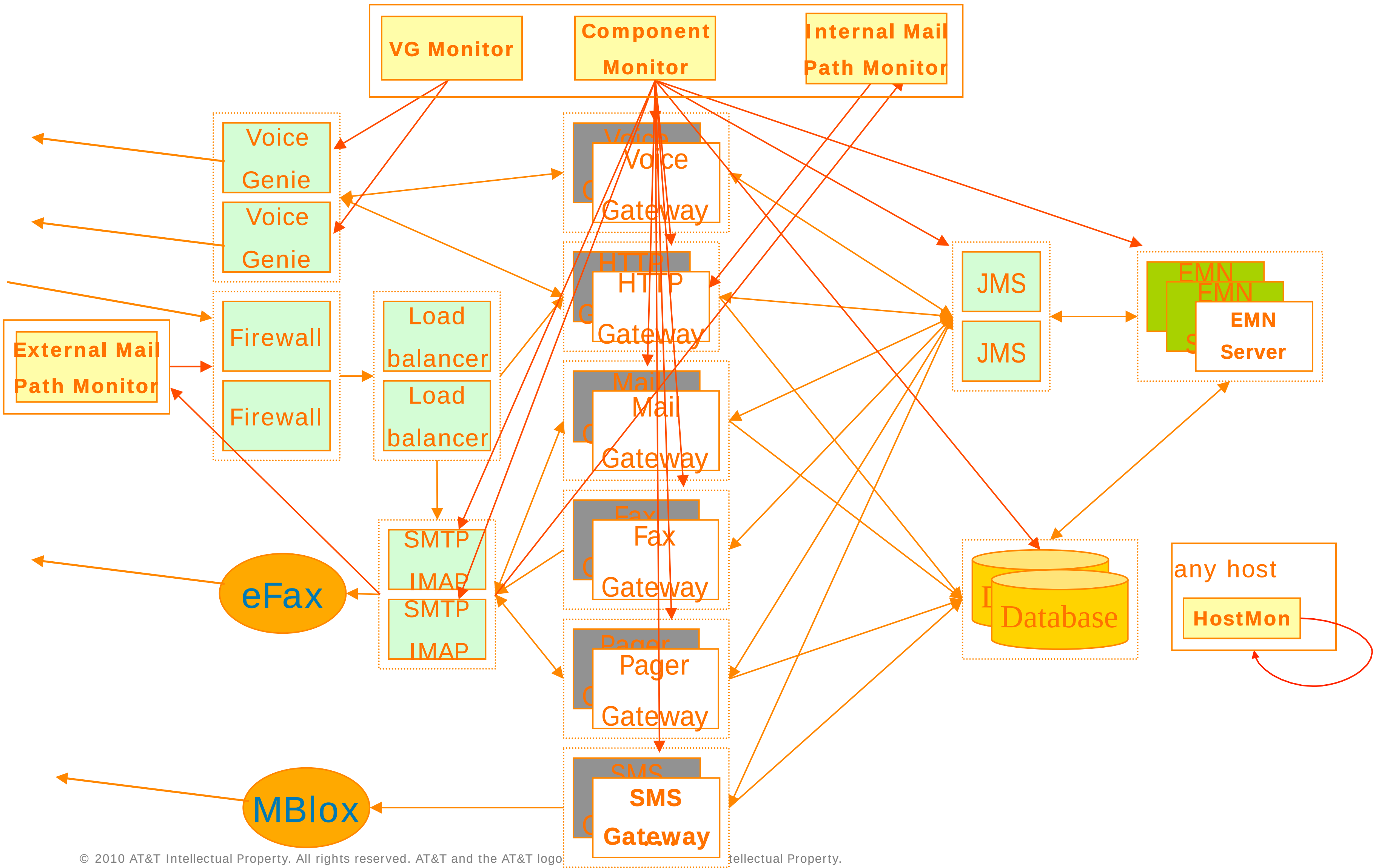
With Robin Chen, Rittwik Jana, the rest of the EMN team at AT&T Labs-Research.

Architecture of Enterprise Messaging Network



MD – Mobile device

Monitoring

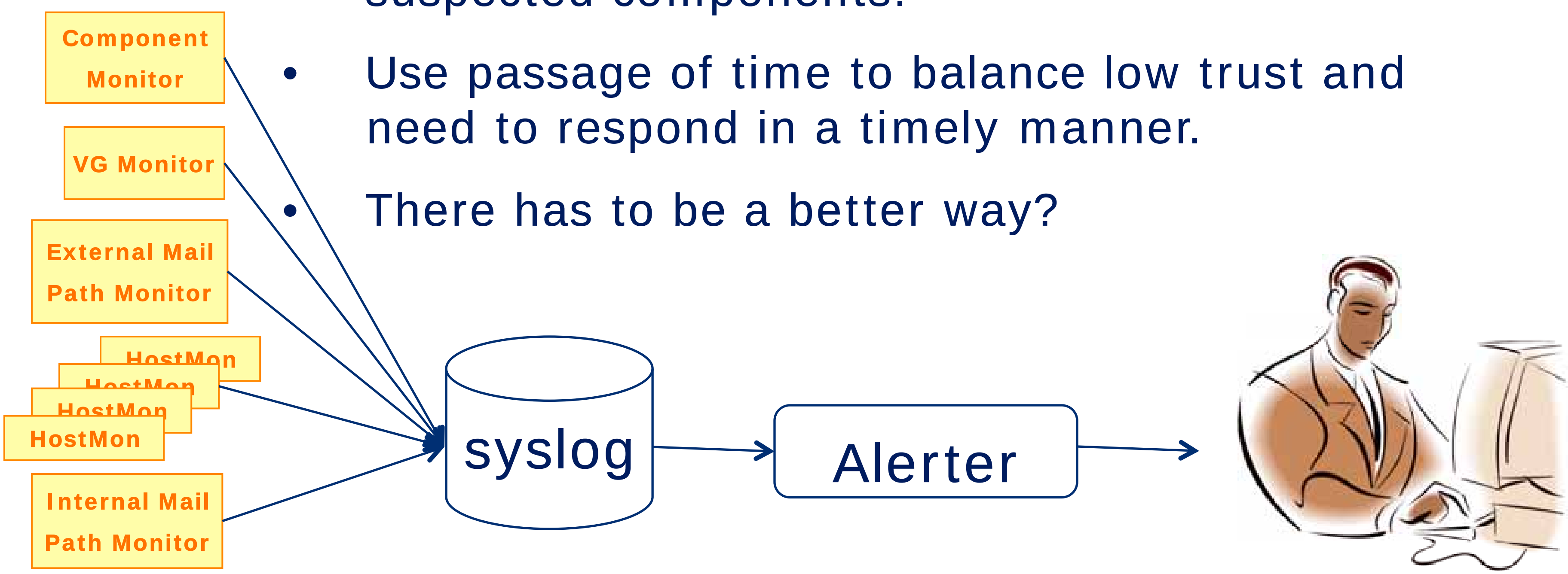


VG Monitor
Component Monitor
Internal Mail Path Monitor
Voice Genie
Voice Genie
External Mail Path Monitor
Firewall
Firewall
Load balancer
Load balancer
Voice Gateway
HTTP Gateway
Mail Gateway
Fax Gateway
Pager Gateway
SMS Gateway
SMTP IMAP
SMTP IMAP
eFax
MBlox
JMS
JMS
EMN
EMN
EMN Server
Database
any host
HostMon

Diagnosis and Alerting

Combine information from different monitors:

- Different reporting frequencies and accuracies.
- Use negative monitor outputs to suspect components, positive outputs to exclude suspected components.
- Use passage of time to balance low trust and need to respond in a timely manner.
- There has to be a better way?



3. Stochastic Model-Driven Diagnosis and Recovery

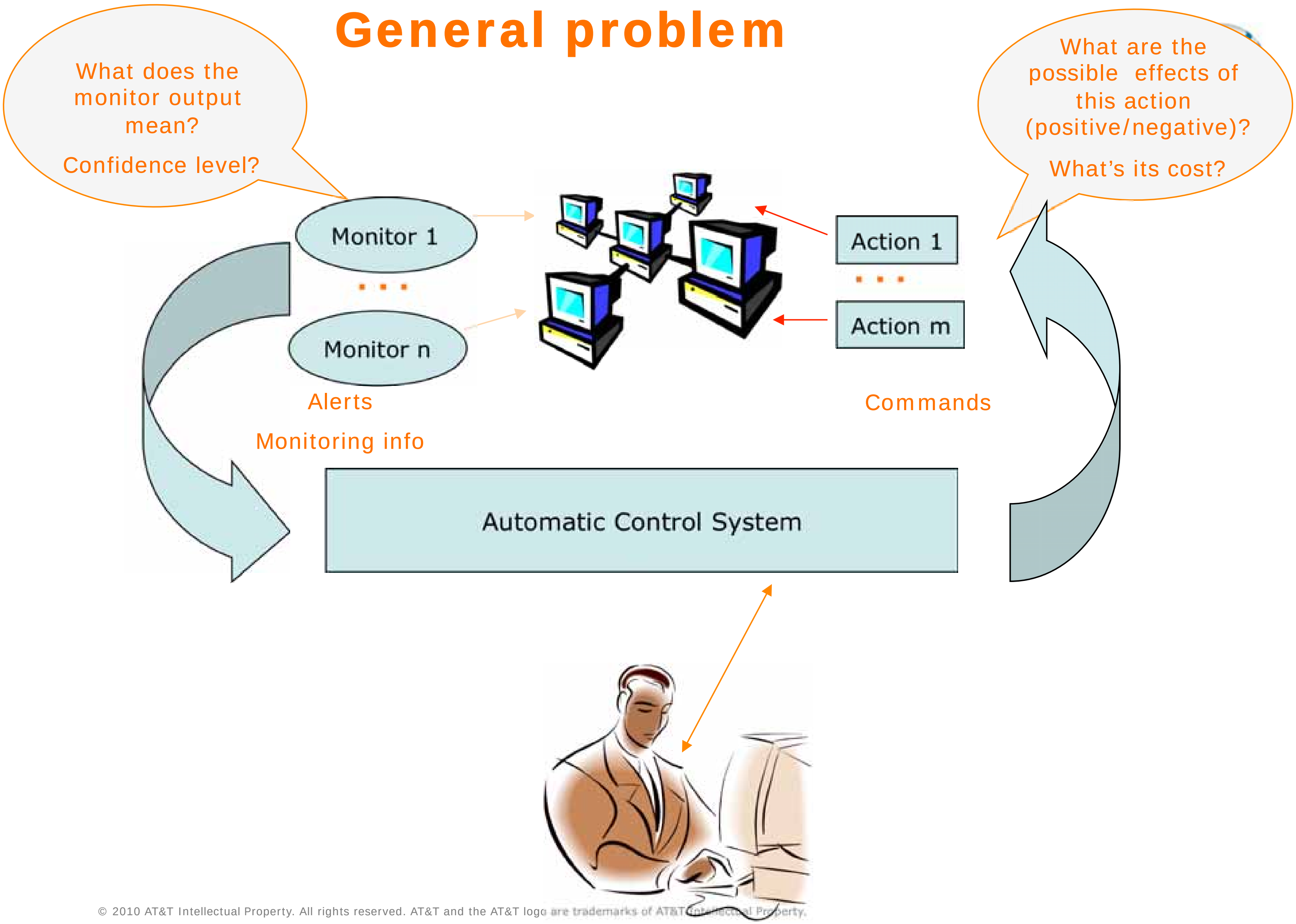
With Kaustubh Joshi (UIUC/AT&T),

Bill Sanders (UIUC), Rick Schlichting (AT&T)

Problem addressed: How to deal with uncertainty in monitoring information and how to choose optimal recovery actions given uncertainty.

“Automatic Model-Driven Recovery in Distributed Systems.” SRDS 2005: 25-38.

General problem



Challenges in System Level Recovery

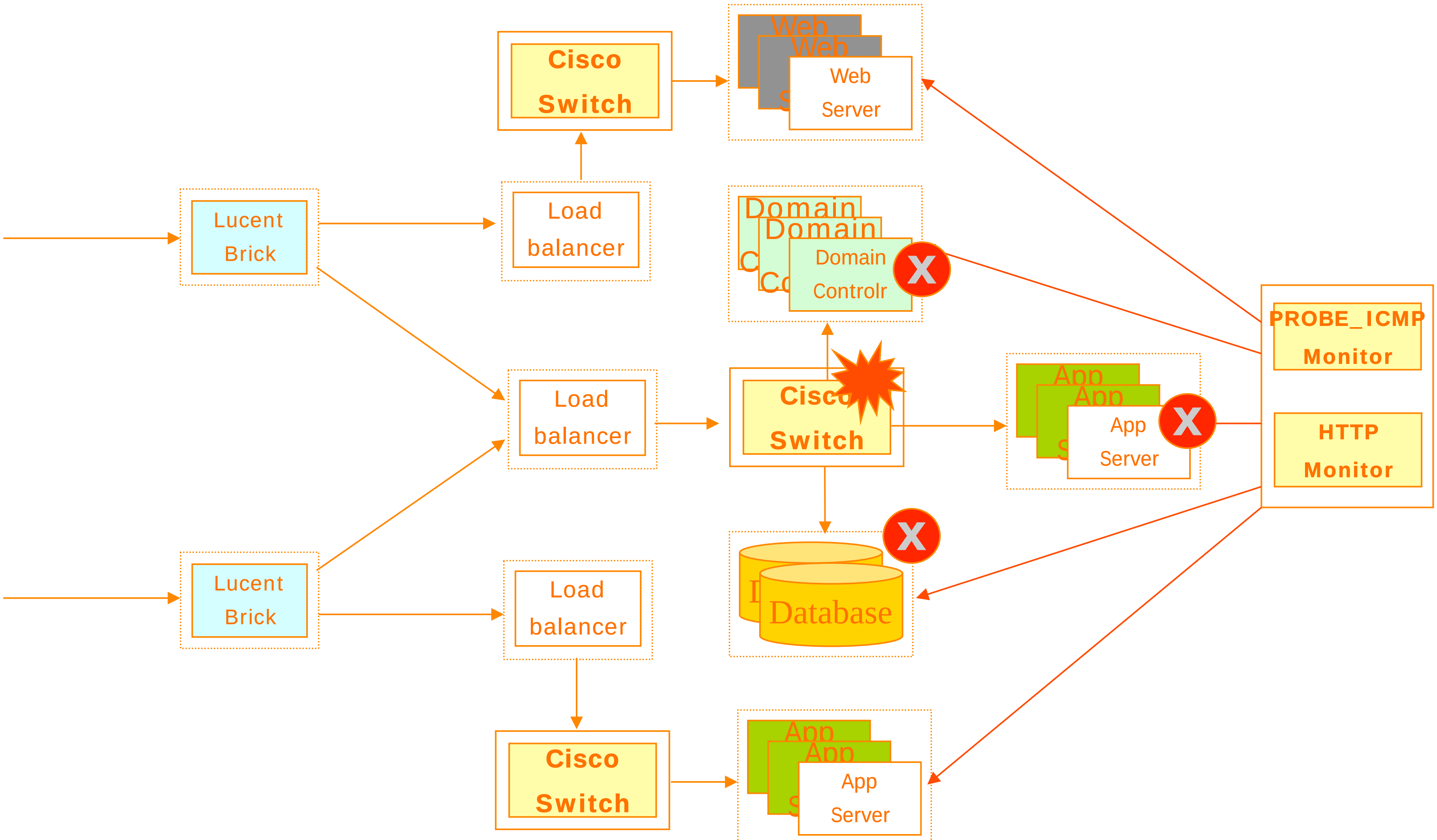
When change is needed, what to do?

- Action: do nothing, restart, fail-over, reconfigure, degrade
- Target: component, subsystem, whole system
- Different effects, costs, benefits
- Need metrics (cost/rewards) to perform automatically
- Operators implicitly use same information

But if we only knew what the problem was …

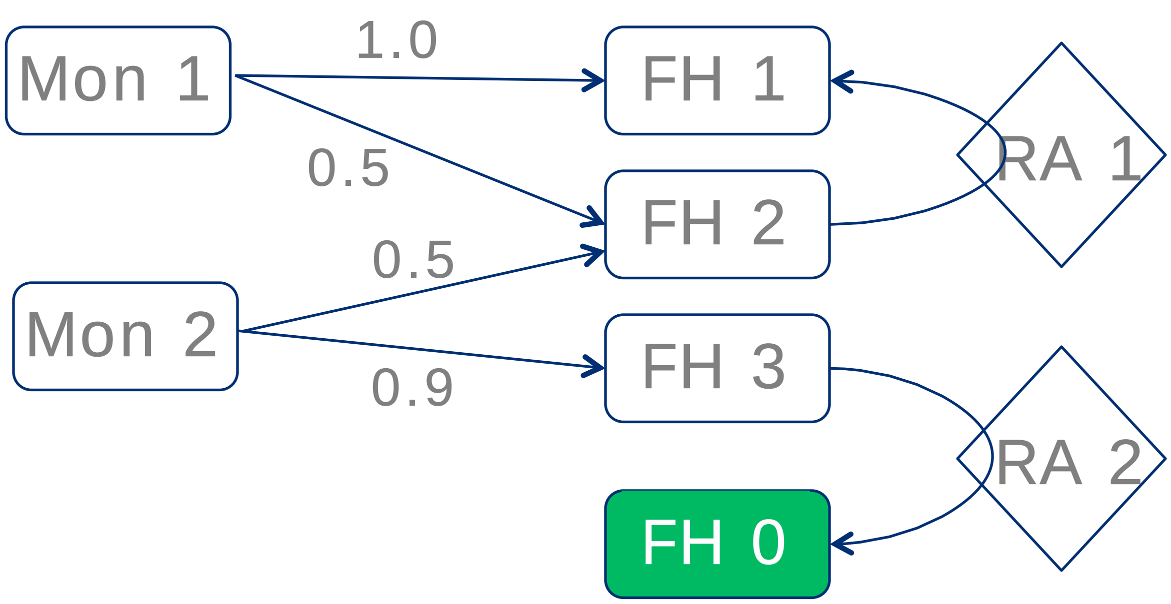
- Monitoring in one tier, fault in another
- Poor localization, false positives and negatives
- Each monitoring technique has different strengths, limits
- Result: uncertainty about true system state

Monitor Coverage Models: Asset monitors



Solution strategy – design time

1. Identify possible fault modes in the system (which components may fail and how) => fault hypotheses.
2. Characterize each existing monitor m in terms of how likely it is to detect each fault hypothesis h: monitor coverage
3. Characterize each recovery action in terms of its impact on fault hypotheses.



Solution strategy - runtime

When at least one monitor reports an error:

1. Combine information from all monitors and prior knowledge about failure rates using Bayesian estimation to determine most likely fault hypotheses.
2. Choose the **optimal** recovery actions most likely to fix the situation*. Execute action(s).
3. Re-execute monitors. If still an error, remember which recovery action(s) were taken and repeat from 1.
4. If it becomes clear that the actual fault scenario is unknown for the automatic recovery system, the operators are alerted.

*Note: Different algorithms of different complexity are possible for choosing optimal recovery action. Current set includes simple one step optimization algorithm and a multi-step optimization algorithm based on Partially Observable Markov Decision Processes (POMDPs).

Example: AT&T Hosting Alarm Suppression

AT&T Managed Hosting Services

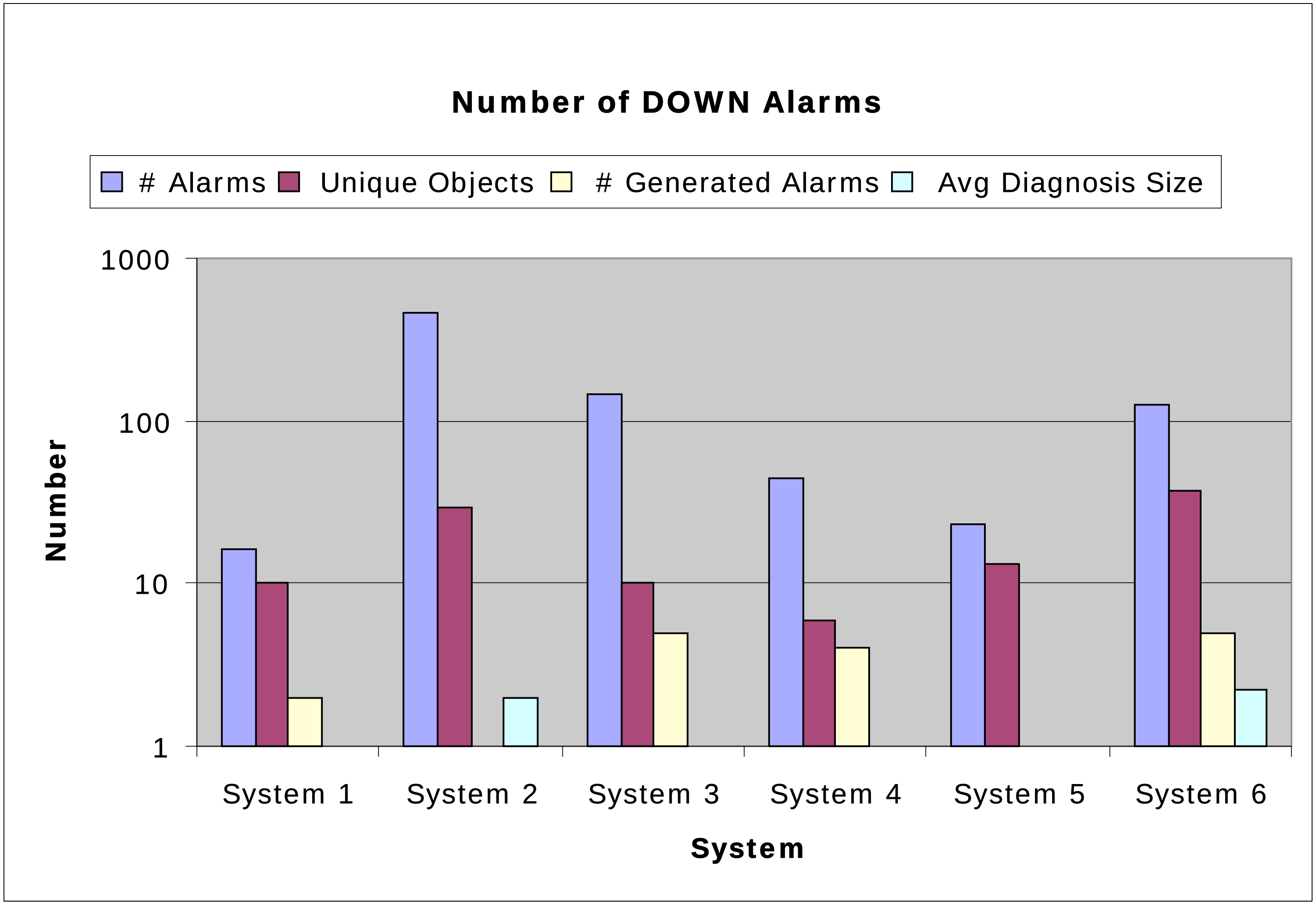
- Data centers running customer applications
- Different configurations, similar components
- Automated common monitoring infrastructure

Monitoring generates lots of alarms

- 1.67 million alarms in example month
- 483k critical alarms
- Alarms with multiple competing causes
- Multiple alarms with common cause

AT&T Hosting Alarm Suppression Results

	System 1	System 2	System 3	System 4	System 5	System 6
# Nodes	55	33	48	25	188	135



Benefits and Limitations

Benefits

- Separation of concerns: monitoring and recovery.
- Sequential recovery a natural way to deal with mistakes
- Ability to look multiple time-steps ahead
  - knows when to wait for additional information
  - can use outcomes of recovery actions to make better choices
- Formal framework
  - strong guarantees about stability and goodness of adaptation

Limitations

- Model Based:

  Models can be wrong to start with or become wrong due to changes in the system.

4. Diagnosis in VoIP Systems

With Kaustubh Joshi (AT&T), Soila Pertet (CMU), Scott Daniels (AT&T)
+ Priya Narasimhan (soon).

Problem addressed: How to diagnose failures never (or rarely) seen before.

Motivation

Business VoIP services

- 10+ service types, 200+ network elements
- Millions of calls per day, growing rapidly

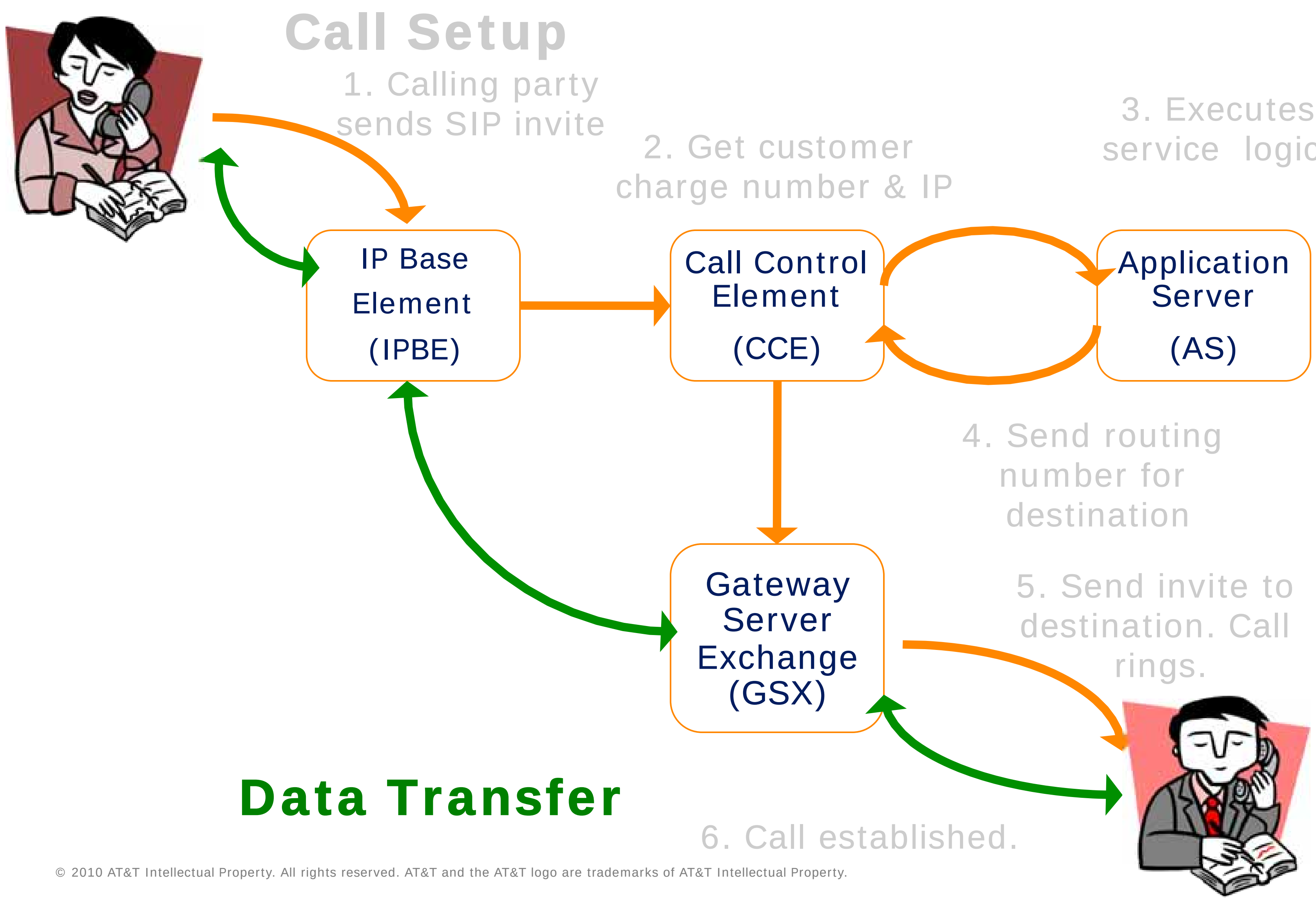
Faults occur continuously in the system

- Minor incidents + major incidents, e.g., failed upgrades, can increase fault rate
- Faults cause failed calls (blocked or cut-off calls)

Research question

- How to diagnose faults that have never happened before?
- Faults due to combinations of unexpected events?

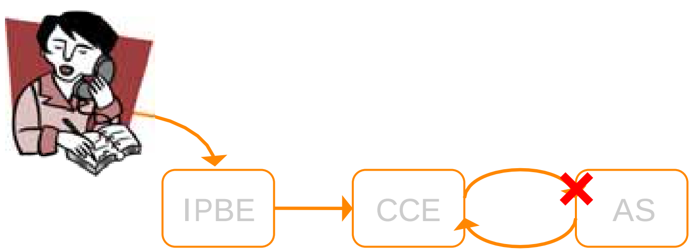
VoIP SIP Call Flow



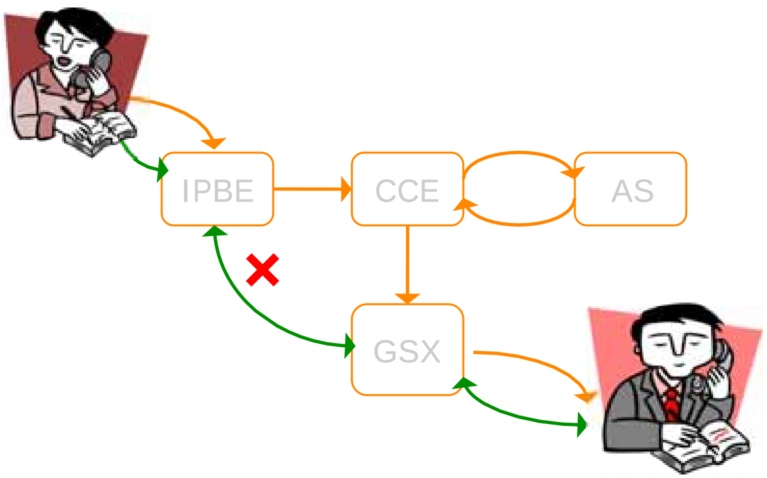
VoIP Call Detail Records (CDRs)

Network elements record call outcome in CDRs

- Timestamp, name of network element, service type
- Telephone numbers, customer IP addresses
- Outcome of call (successful, blocked, or cut-off call)



Blocked calls: Fail during setup



Cut-off calls: Failure after setup

Hierarchical Bayesian Algorithm

Adopt a technique from software debugging literature:

- Liu, C., Lian, Z., and Han, J. "How Bayesians Debug". In *Proc. 6th IEEE Int. Conf. on Data Mining.* Dec. 2006.

Generic algorithm: Uses very little domain knowledge.

Operates on CDR "attributes"

- Service types, defect codes, network element names
- Customer IP addresses
- Easily extended to include additional call attributes
  - e.g. software versions, QOS data

Identifies call attributes most correlated with failed calls.

"Truth table" Call Representation

Successful Call

- ny4ny01sdh
- ph4pa0102bap
- ny4ny02gh
- ....

Failed Call

- ny4ny01sdh
- at4ga03wap
- ny4ny02gh
- ....

ny4ny01sdh	ph4pa0102bap	at4ga03wap	ny4ny02gh	Call Outcome
1	1	0	1	SUCCESS
1	0	1	1	FAIL

Solution Architecture

Incoming CDR Stream

(Labeled, Correlated)

Attribute Generation

(Domain Specific)

Diagnosis

(Domain Independent)

Diagnosis Presentation

(Domain Specific)

Attribute Generation

Most attributes picked directly from CDR

- Network element name, service, defect codes, success codes, customer IP

Add aggregate attributes

- e.g., at4ga*wap

Discard attributes that are not useful to diagnosis

- E.g., success codes

Opportunity to add additional features

- E.g., software version of all nodes call passes through
- Node utilization information (e.g., server overloaded)

We have about 10,000 attributes (naïve selection)

Suspect Attribute Identification

In each call:

- Assume each attribute has a stable but unknown occurrence probability.
- Reflects service volume, call distribution, routing, etc.

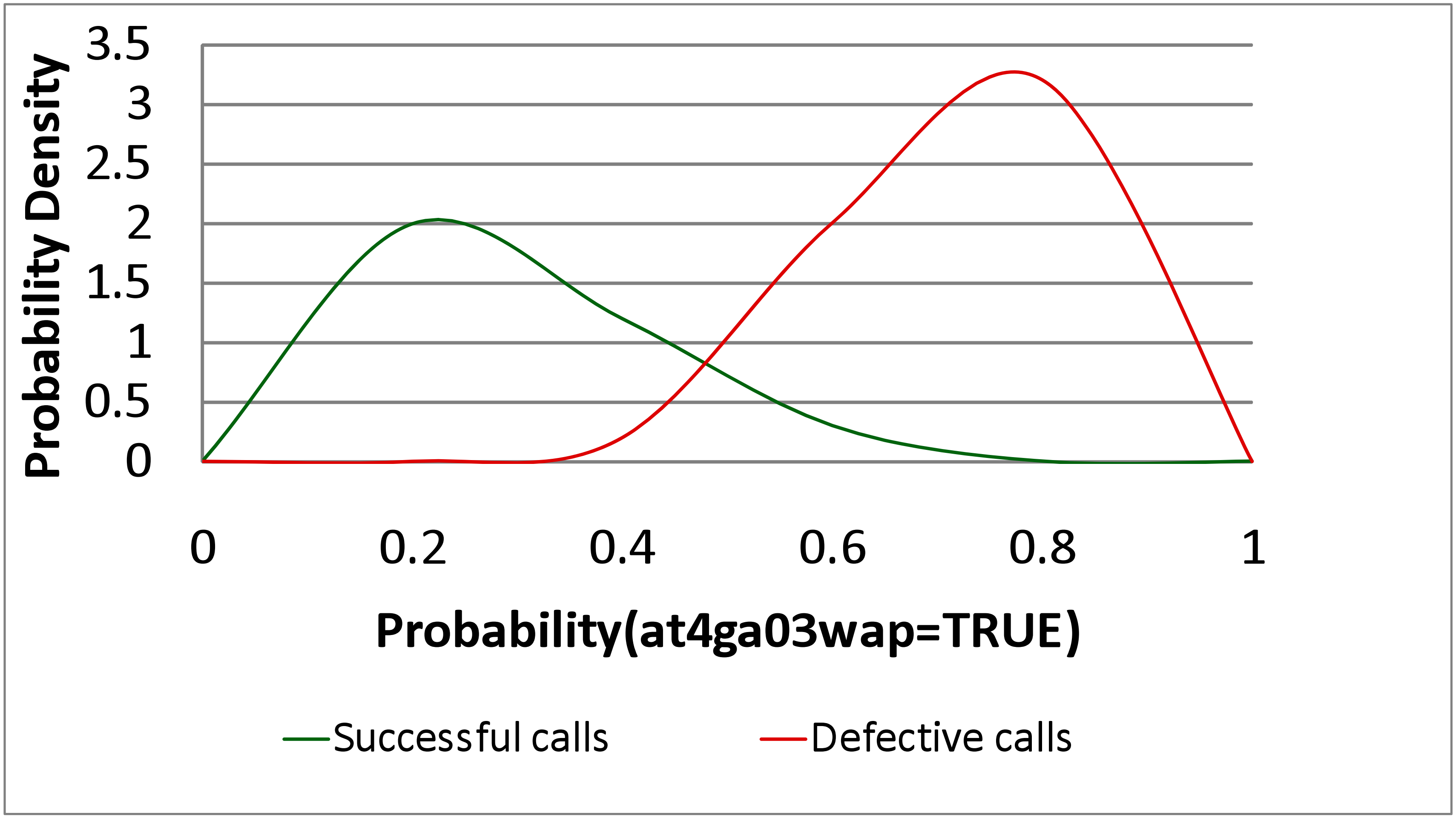
Bayesian estimation:

- Construct failure/success distributions for attribute occurrence probabilities:
  - P[Attribute occurs in failed calls],
  - P[Attribute occurs in successful call]
- Each CDR updates either failure or success distribution

Distribution divergence

- Compute the difference between success and failure distributions
- Attribute with largest divergence is chosen as the suspect attribute

Success and Failure Attribute Distributions



Estimating Attribute Occurrence Distributions

Initial value of attribute occurrence probability

- Uniform[0,1]

Bayesian update on incoming CDR

- Success CDR: Bayes update to success distribution
- Failure CDR: Bayes update to to failure distribution

Result is a Beta distribution

- Uniform distribution → Bayes rule → Beta distribution
- Beta distribution → Bayes rule → Beta distribution

After all CDRs processed:

- Success distribution:
  - Beta(1 + Num good calls with attribute, 1 + Num good calls w/o attribute)
- Failure distribution:
  - Beta(1 + Num bad calls with attribute, 1 + Num bad calls w/o attribute)

Comparing Success and Failure Distributions

Kullback Leibler Divergence

- Information theoretic distance between two probability distributions
- Given fail and success distributions P and Q with densities p(x) and q(x)
- $$KL(P||Q) = \int_{-\infty}^{+\infty} p(x) \log \frac{p(x)}{q(x)} dx$$

Closed form for Beta distributions: KL(Beta(a,b) || Beta(c,d)) =

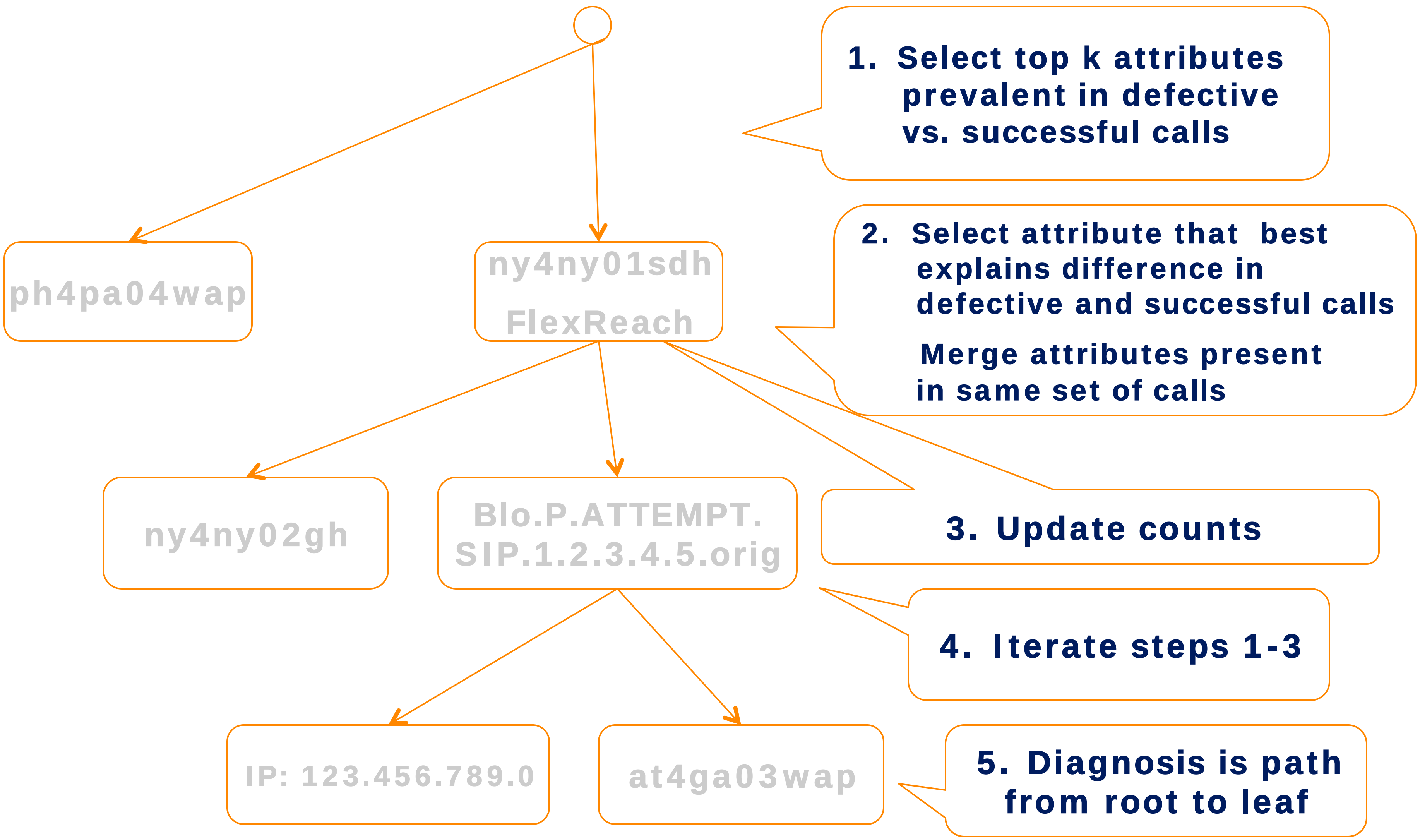
$$\ln \frac{\mathcal{B}(c,d)}{\mathcal{B}(a,b)} + (a-c)[\Psi(a) - \Psi(a+b)] + (b-d)[\Psi(b) - \Psi(a+b)]$$

- Where $\mathcal{B}$ and ψ are standard beta and digamma functions
- a/b: 1 + Number of failed calls with/without attribute
- c/d: 1 + Number of successful calls with/without attribute

Easy to compute and incrementally update:

- Only requires CDR counts

Iterative Bayesian approach



Current/future work

Output formatting:

- How to present the diagnosis output in a form that is useful for system operators (tree format not convenient).

Adding some domain knowledge:

- Which attributes to omit,
- Which attributes are identical/redundant,
- Which defect codes really mean the same thing (e.g., various timeouts).
- ...

Validation of diagnosis output with domain experts.

Addition of other information sources.

Real-time analysis.

5. Conclusions

- Lots of opportunities and unsolved problems for failure diagnosis in enterprise systems.
- Depending on the type of data available, different techniques will be required.
- Future enterprise systems should be designed with diagnosability in mind!