Solaris 10 Predictive Self-Healing: Fault Management

Mike Shapiro

mws@sun.com http://blogs.sun.com/mws/

Solaris Kernel Development, Sun Microsystems

An Ideal Model of Availability

- Easy to compose and deploy a solution of known availability and measure it continuously
- System actively prevents, diagnoses, and recovers from failures, maximizing solution availability
- Administrators are presented with meaningful, guided interactions when intervention required
- Vendor uses data-driven feedback loop to improve product quality, make informed business decisions

Self-Healing in Solaris 10

- Fault Manager
  - automated diagnosis and isolation of hardware faults
  - new structured log files and tools for telemetry data
  - live diagnosis updates without reboots
  - standardized fault messaging
- Service Manager
  - integrated, automatic restart of failed software services
  - automated, guided troubleshooting for failed services
  - faster boot, improved disaster recovery, security, more ...
- Messages linked to new knowledge article web site

Self-Healing in Solaris 10 (cont.)

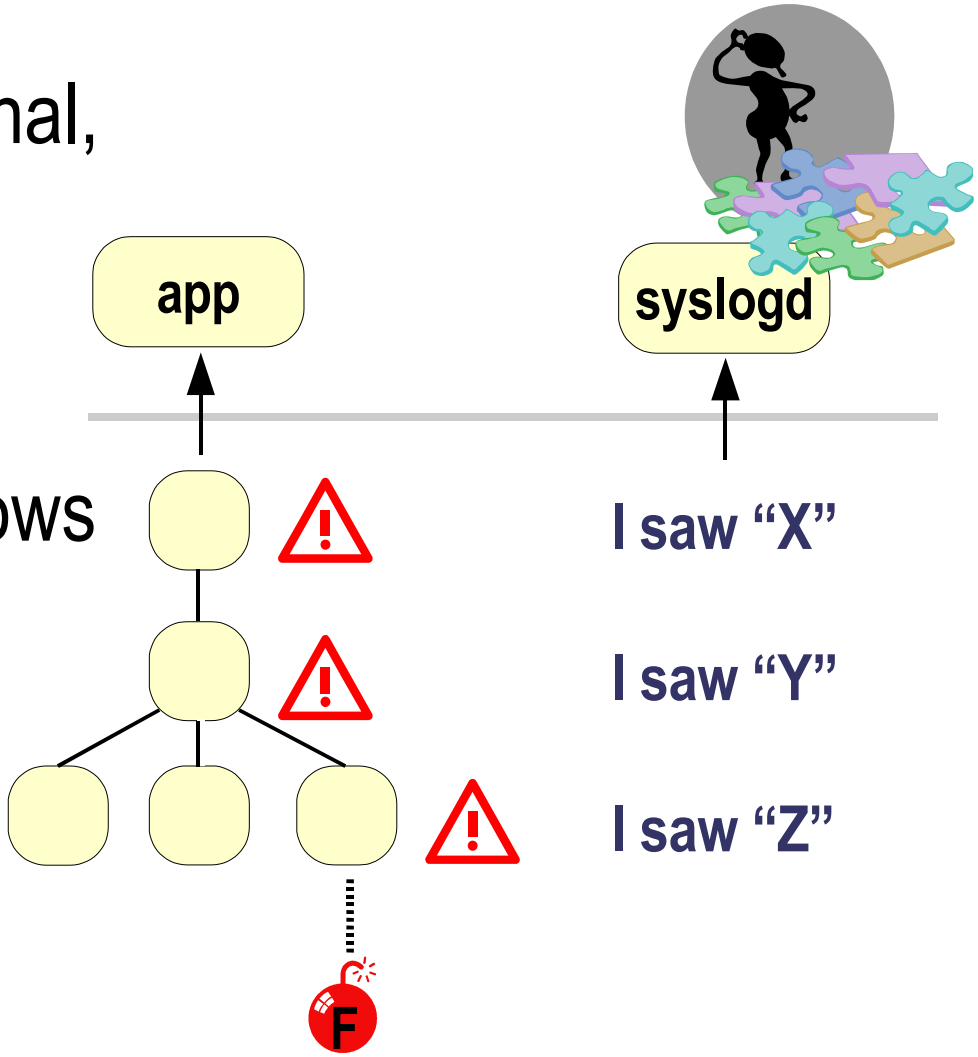
- Diagnosis for UltraSPARC III, IV CPU/Memory
- Automatic faulty CPU, memory page retire
- Automatic CHS updating on high-end servers
- Diagnosis for UltraSPARC PCI HBAs
- Improved resilience for all PCI I/O failures
- Basic unbundled support at FCS:
  - Explorer will capture new Fault Manager logs
  - SunMC can generate alerts for diagnosis results
  - SunVTS updated to work properly with FMA

And more coming in S10 updates …

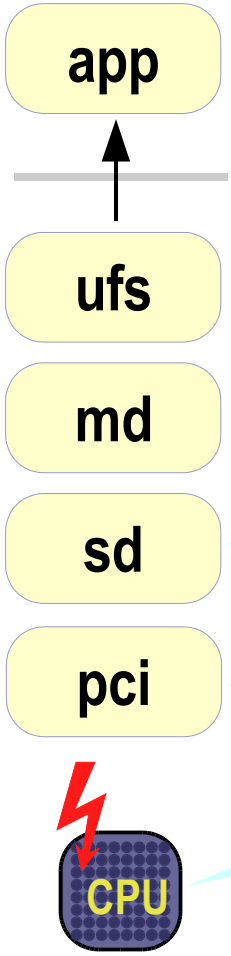
- x86/Opteron CPU/Memory support
- x86/Opteron PCI, PCI-X support
- x86/Opteron/SPARC PCI-E support
- Generic SNMP trap/MIB support
- SMART disk support, connections to ZFS
- Links, APIs for management apps, scripts
- Hardened disk, network leaf drivers
- FM support “out of the box” on future platforms
- Locally hosted knowledge article web

Legacy Model: Error Centric

- **Error** – an incorrect signal, datum, or result
- Observation that is a *symptom* of a fault
- Legacy system only knows how to report errors
- Diagnosis and reaction left up to humans



For example ...



- Each component captures data, tries to handle the error, and produces a syslog message
- Humans try to diagnose *fault*, *impact*, and appropriate *corrective action*

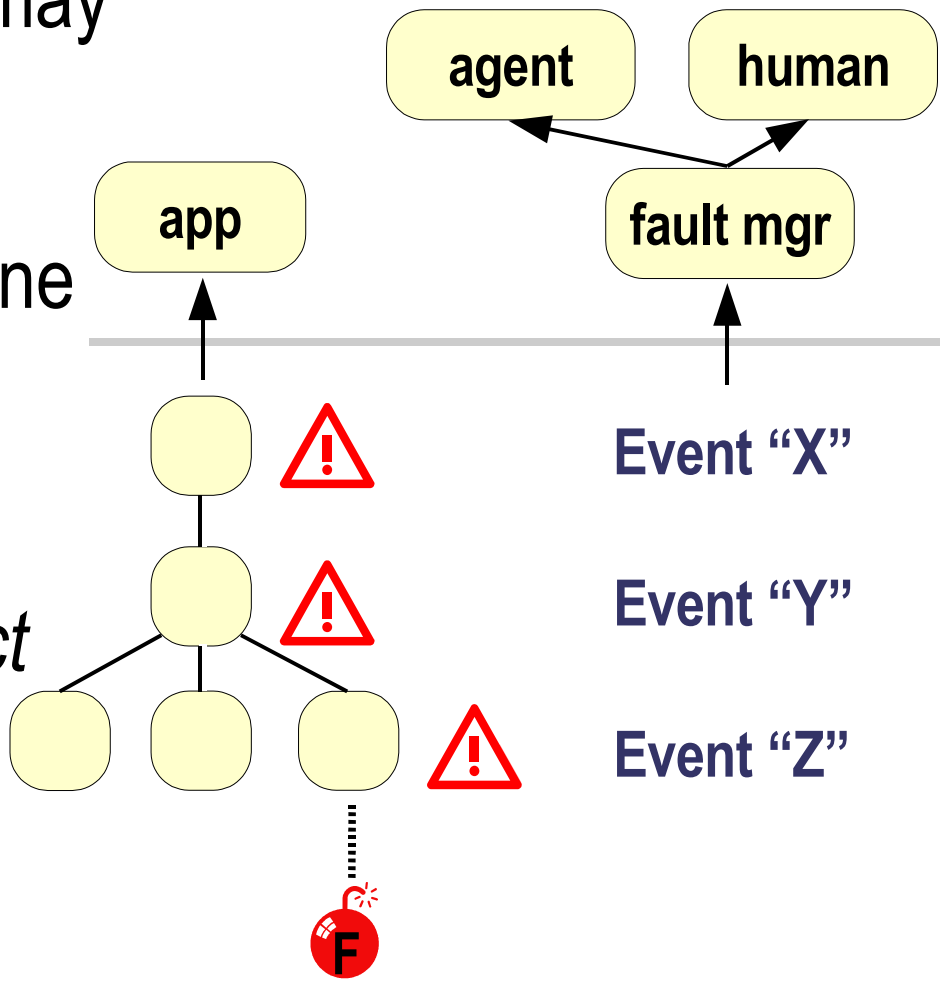
```
WARNING:/io-unit@fe0200000/sbi@0,0/dma@0,81000/
esp@0,8000000 (esp0): Connected command timeout
for Target 0.0
```

```
NOTICE: correctable error detected by pci0
(upa mid 1f) during DVMA read transaction
AFSR=40f40000.1f800000 AFAR=00000000.a25b4000 ...
```

```
[AFT0] errID 0x0000004d.23105c04 Corrected Memory Error
on U1004 is Intermittent
[AFT0] errID 0x0000004d.23105c04 ECC Data Bit 14 was in
error and corrected
```

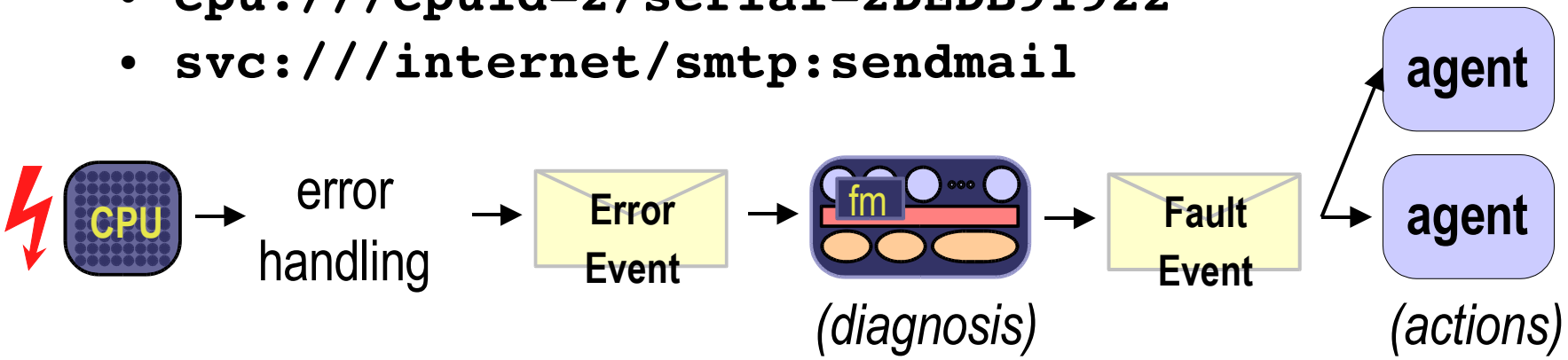
New Model: Automated Diagnosis

- **Fault** – a problem that may produce errors
- A fault is diagnosed or predicted based upon one or more errors or other observations
- Something we can associate with an *impact* and a *corrective action*
- Diagnosis software automates the steps



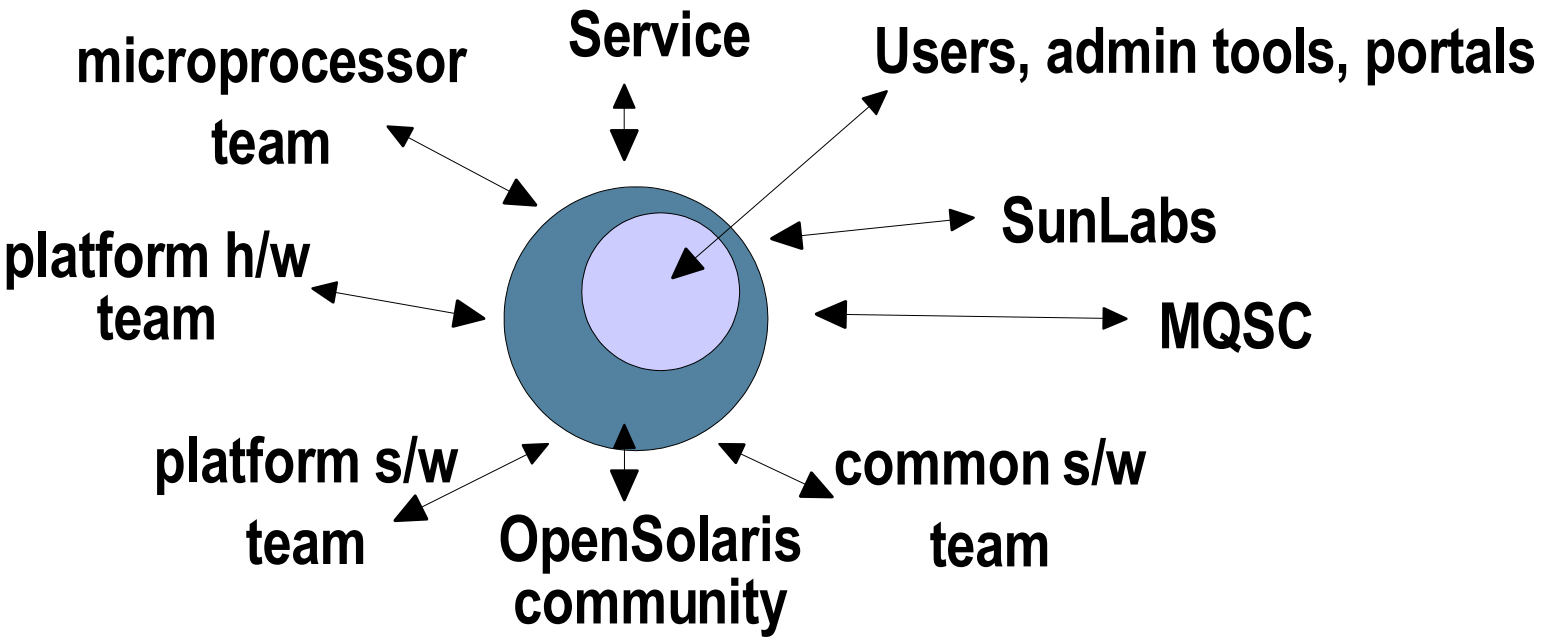
Event Telemetry Flow

- Protocol describes common aspects of error events and fault events and an encoding of an event
- All events defined and versioned in Sun event registry
- Subsystems define appropriate classes (e.g. `ereport.cpu.UltraSPARC-III.ce`)
- Resources are named using “FMRIs” :
  - **`cpu:///cpuid=2/serial=2DEDB91922`**
  - **`svc:///internet/smtp:sendmail`**



Event Registry (http://events.central)

- Docs and interfaces for collaborative FMA work
- Used to build customer knowledge web nightly
- Just a workspace underneath: working on migrating content, process, and tools to opensolaris.org



Fault Manager: fmd(1M)

- Daemon that multiplexes event telemetry to subscribing modules and manages the collection of:
  - diagnosis engines
  - response agents
  - system management agents
  - resource cache
  - telemetry logs (error log and fault log)
- Two-phase commit for events, checkpoints activity
- Observable using new administrative tools
- Providing common APIs, live update capability

Self-Healing Administration Model

- **fmd** dispatches telemetry events to appropriate diagnosis engine based upon subscriptions
- Events accumulated into *cases*, named by a *UUID.* UUID identifies this problem in the enterprise.
- Cases are *solved* by associating one or more suspect fault events that explain the symptoms
- Classes of suspects correspond to a unique *MSGID.* MSGID can be used to select message, web article.
- When solution is ready, a **list.suspect** event is published to response and management agents

Syslog Messages Agent

- Prints a localized diagnosis message to the console and to /var/adm/messages through syslogd(1M)
- New messages can be optionally set to LOCAL[0-7]

SUNW-MSG-ID: SUN4U-8000-AC, **TYPE**: Fault, **VER**: 1, **SEVERITY**: Major

EVENT-TIME: Thu Feb 26 18:08:26 PST 2004

PLATFORM: SUNW,Sun-Fire-V440, **CSN**: -, **HOSTNAME**: mix

SOURCE: cpumem-diagnosis, **REV**: 0.1

EVENT-ID: 322fe6d5-fe14-6a73-b802-cc6c30b2afcd

DESC: The number of errors associated with this CPU has exceeded acceptable levels. Refer to http://sun.com/msg/SUN4U-8000-AC for more information.

AUTO-RESPONSE: An attempt will be made to remove the affected CPU from service.

IMPACT: Performance of this system may be affected.

REC-ACTION: Schedule a repair procedure to replace the affected CPU.

Use **fmdump -v -u** <*EVENT-ID*> to identify the component to be replaced.

http://sun.com/msg/

- Customer knowledge article web site provides examples, impact, and repair procedures
- Updated constantly with latest best practices
- Links to information on latest self-healing features, updates, plans, etc.
- No passwords, etc. - free access
- Plan to permit customers to subscribe and cache locally, directing Solaris messages to custom site
- Easy to connect to web portals, management software, custom management webs and tools

Example: Correctable L2$ Errors

- Multiple L2$ errors seen on the same CPU, same line, not secondary effects, short time span:

SUNW-MSG-ID: **SUN4U-8000-AC**, **TYPE**: Fault, **VER**: 1, **SEVERITY**: Major
EVENT-TIME: Tue Jun 14 21:47:45 PDT 2005
PLATFORM: SUNW,Sun-Fire-V240, CSN: -, HOSTNAME: cygnus
SOURCE: cpumem-diagnosis, REV: 1.4
EVENT-ID: **21653d14-cba7-6535-ccef-ea7a22436149**
DESC: The number of errors associated with this CPU has exceeded acceptable levels. Refer to **http://sun.com/msg/SUN4U-8000-AC** for more information.
AUTO-RESPONSE: An attempt will be made to remove the affected CPU from service.
IMPACT: Performance of this system may be affected.
REC-ACTION: Schedule a repair procedure to replace the affected CPU. Use fmdump -v -u <EVENT_ID> to identify the CPU.

```
# psrinfo
0       on-line   since 06/08/2005 18:36:09
1       faulted   since 06/14/2005 21:47:45
```

Example, Cont.

```
# fmdump -v -u 21653d14-cba7-6535-ccef-ea7a22436149
TIME                 UUID                                 SUNW-MSG-ID
Jun 14 21:47:45.6518 21653d14-cba7-6535-ccef-ea7a22436149 SUN4U-8000-AC
  100%  fault.cpu.ultraSPARC-IIIi.l2cachedata
        FRU: hc:///component=MB
         rsrc: cpu:///cpuid=1/serial=22CD205067

# fmadm faulty
   STATE RESOURCE / UUID
-------- ------------------------------------------------------------------
 faulted cpu:///cpuid=1/serial=22CD205067
         21653d14-cba7-6535-ccef-ea7a22436149
```

Example, Cont.

```
#  fmdump -e -V -u 21653d14-cba7-6535-ccef-ea7a22436149
TIME                           CLASS
Jun 14 2005 21:47:45.404202876 ereport.cpu.ultraSPARC-IIIi.ucc
nvlist version: 0
        class = ereport.cpu.ultraSPARC-IIIi.ucc
        ena = 0x1fab8bd751200401
        detector = (embedded nvlist)
        nvlist version: 0
                version = 0x0
                scheme = cpu
                cpuid = 0x1
                cpumask = 0x23
                serial = 0x22cd205067
        (end detector)

        afsr = 0x400008000e0
        afar-status = 0x1
        afar = 0x102692c000
        pc = 0x18384
        tl = 0x0
        tt = 0x70
        privileged = 0
        multiple = 0
        syndrome-status = 0x1
        syndrome = 0xe0
        ...
```

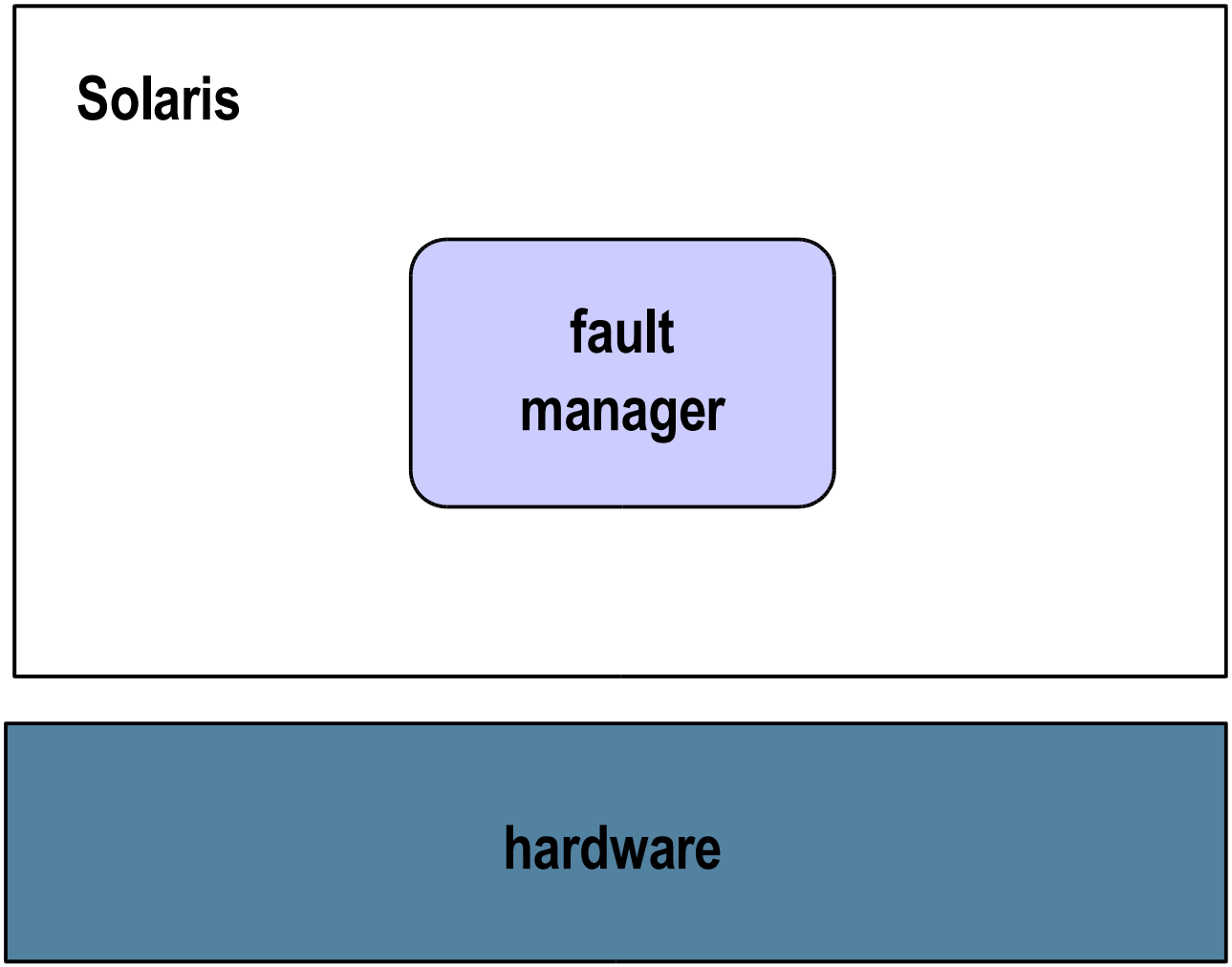
Diagnosis Approaches

- Always start by developing fault tree
- Hand-coded algorithms can be developed in C
- Eversholt language encodes fault tree as a program
- SERD engines can be used for upset/fault situations
- SPRT can be used to predictively analyze variables that are expected to remain constant w/ time
- Many other approaches possible ...

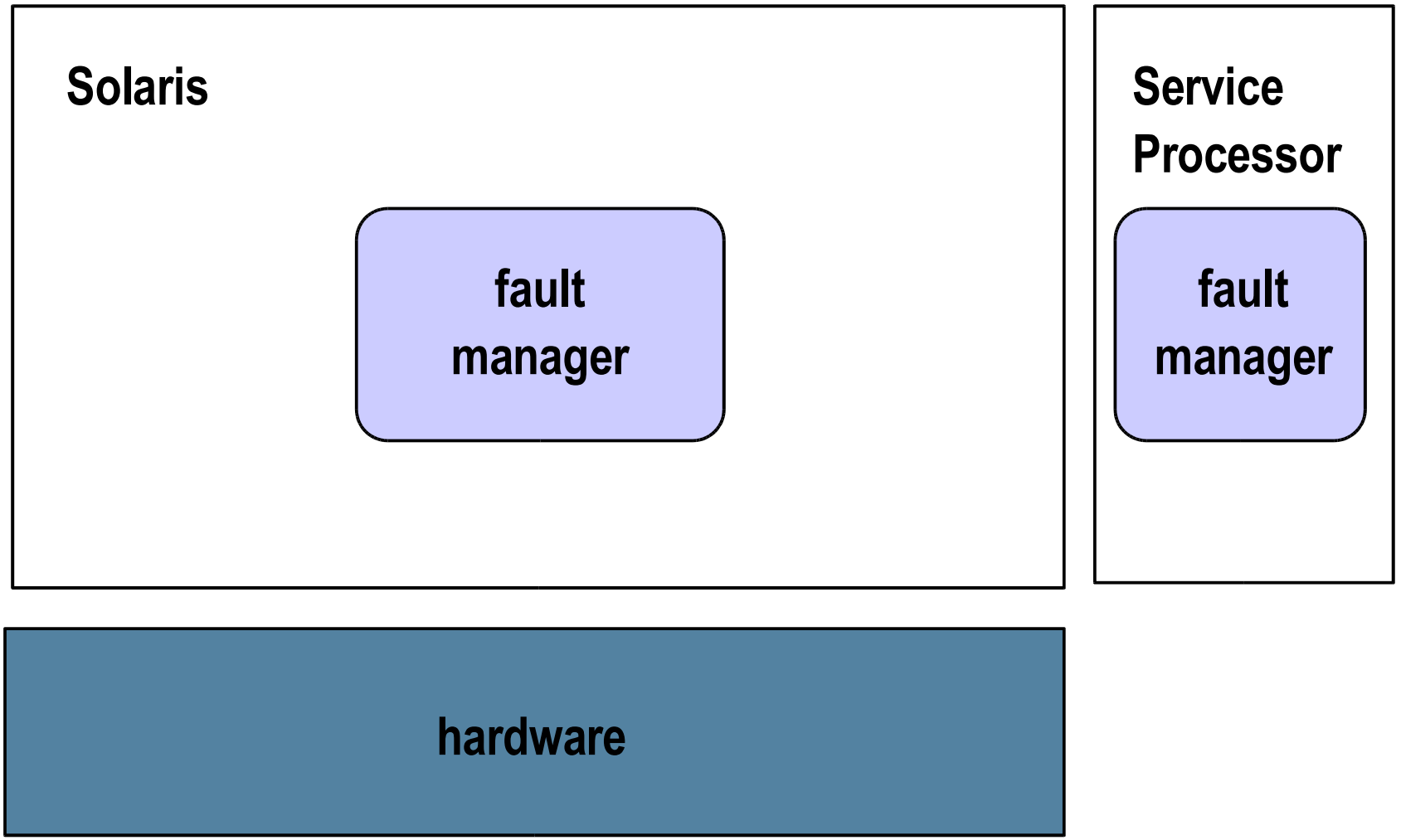
Important Hardware Trends

- PCI-E (point-to-point, much better for diagnosis)
- ChipKill will be ubiquitous on all machines soon
- FBDIMM
  - additional detectors to help diagnosis
  - sparing and mirroring
  - hot-plug and lane fail-over
- More Stuff to Disable
  - sockets, cores, strands, cache lines, lanes, etc.
- Virtualization (making our job much harder)

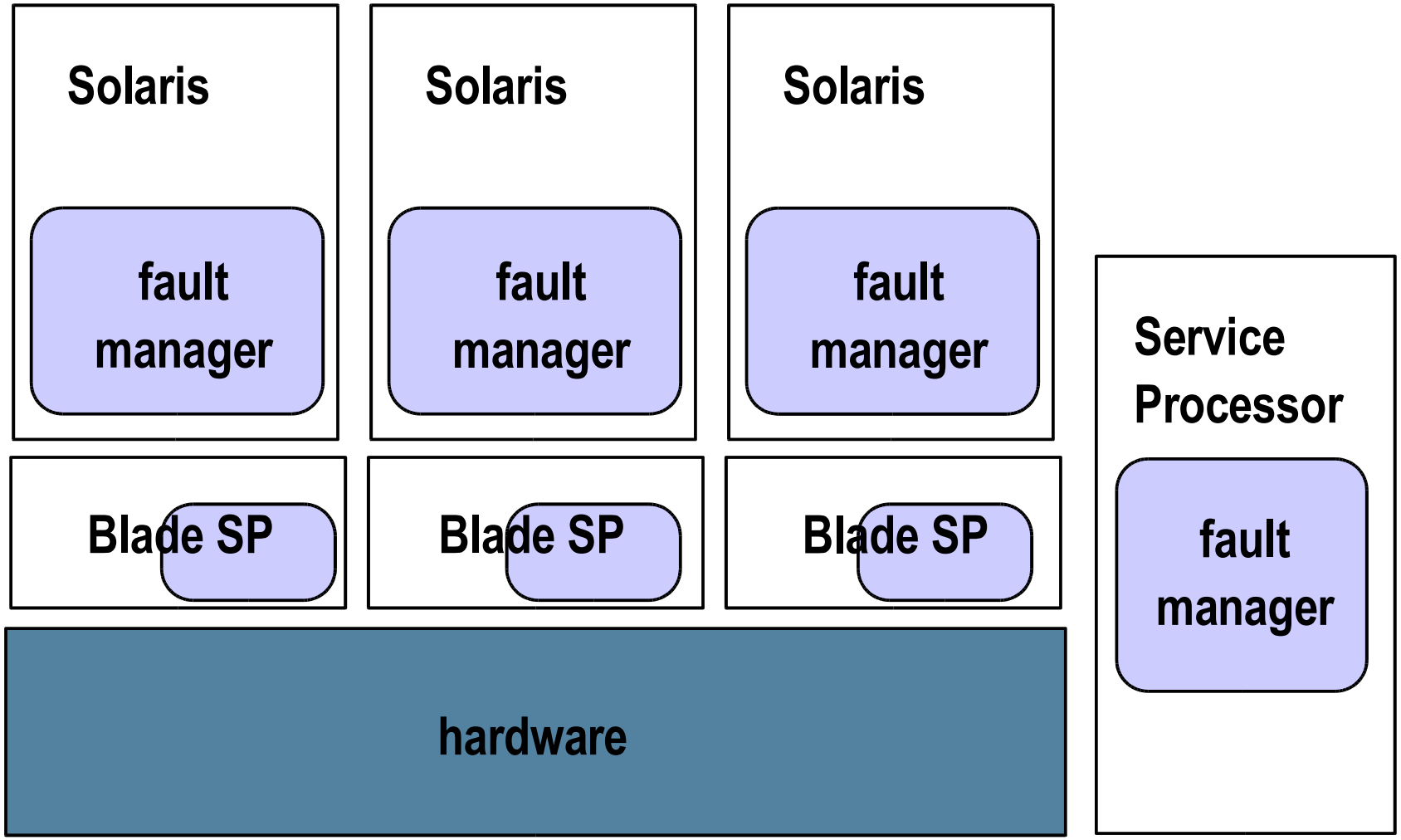
System-Level View: Laptop



System-Level View: Server



System-Level View: Bigger Server



When designing a subsystem ...

- If in userland, describe your subsystem as one or more services with SMF; understand dependencies
- Be prepared for failure: design and test error propagations. Build failure injection software.
- Develop a fault tree (even if on paper). Understand how expected failures will manifest to users.
- Implement appropriate events and FMA content.
- Resources that can contain or be affected by faults should have a faulted state, interface to disable.

Community Ideas

- Contribute improved content for a diagnosis article
- Contribute FMA results to help future development
- Publish FMA events into favorite management tools
- Improve a device driver using protected accesses
- Improve any other type of h/w interaction
- Replace syslog garbage with useful FM interaction
- Convert a legacy or external service to SMF
- Participate in ongoing FM research and development

Examples

- TOD chips that produce inconsistent, wrong results
- Device/nexus spurious interrupt counters/kstats
- Any system message you can't understand
- Anything an administrator can misconfigure
- Anything that can prevent the system from booting
- ...

Solaris 10 Predictive Self-Healing: Fault Management

Mike Shapiro

mws@sun.com http://blogs.sun.com/mws/

Solaris Kernel Development, Sun Microsystems