SOLARIS BOOT CAMP II
Migrating a Legacy RC service

Bob Netherton
Solaris Adoption, US Client Solutions
Sun Microsystems

Agenda

- Quick review of Solaris Service Management Facility
- Example using MySQL
- Creating an SMF Manifest
- Exploring Dependencies
- Importing our new SMF manifest
- Checking it all out
- Questions

SMF First Impression

- Solaris has a wealth of innovations
  - Most of these features are optional, valuable – but optional
- You cannot avoid SMF
  - You experience it on first boot (loading service definitions)
  - /etc/init.d is significantly smaller than in Solaris 9
  - Where did all of those service scripts go ?
- Legacy methods still work
  - Sequencing is very interesting as you migrate services
- Argh!!! - Why did you do all this to me ?

SMF First Impression

- SMF seems more complicated than it really is
- Migrate a legacy RC service to be immersed
  - The first one seems hard
  - The second one is a snap
- SMF rocks!

Terms and Definitions

Service

A long lived software object with a well-defined state, error boundary, definition of start and stop, and relationship to other services. A service is often critical to operation of system or fulfillment of business objectives.

Service Management

- service delivery
- administrative interaction
- management of service by system

Predictive Self-Healing

Solaris Fault Manager (FMA)

- Solaris Fault Manager provides detection and diagnosis of errors, leading to isolation or deactivation of faulty components and precise, accurate administrative messaging.

Solaris Service Manager (SMF)

- smf(5) makes Solaris services self-healing.

Hardware faults which previously caused system restart are now isolated to the affected services. Services are also automatically restarted in the face of hardware and software faults.

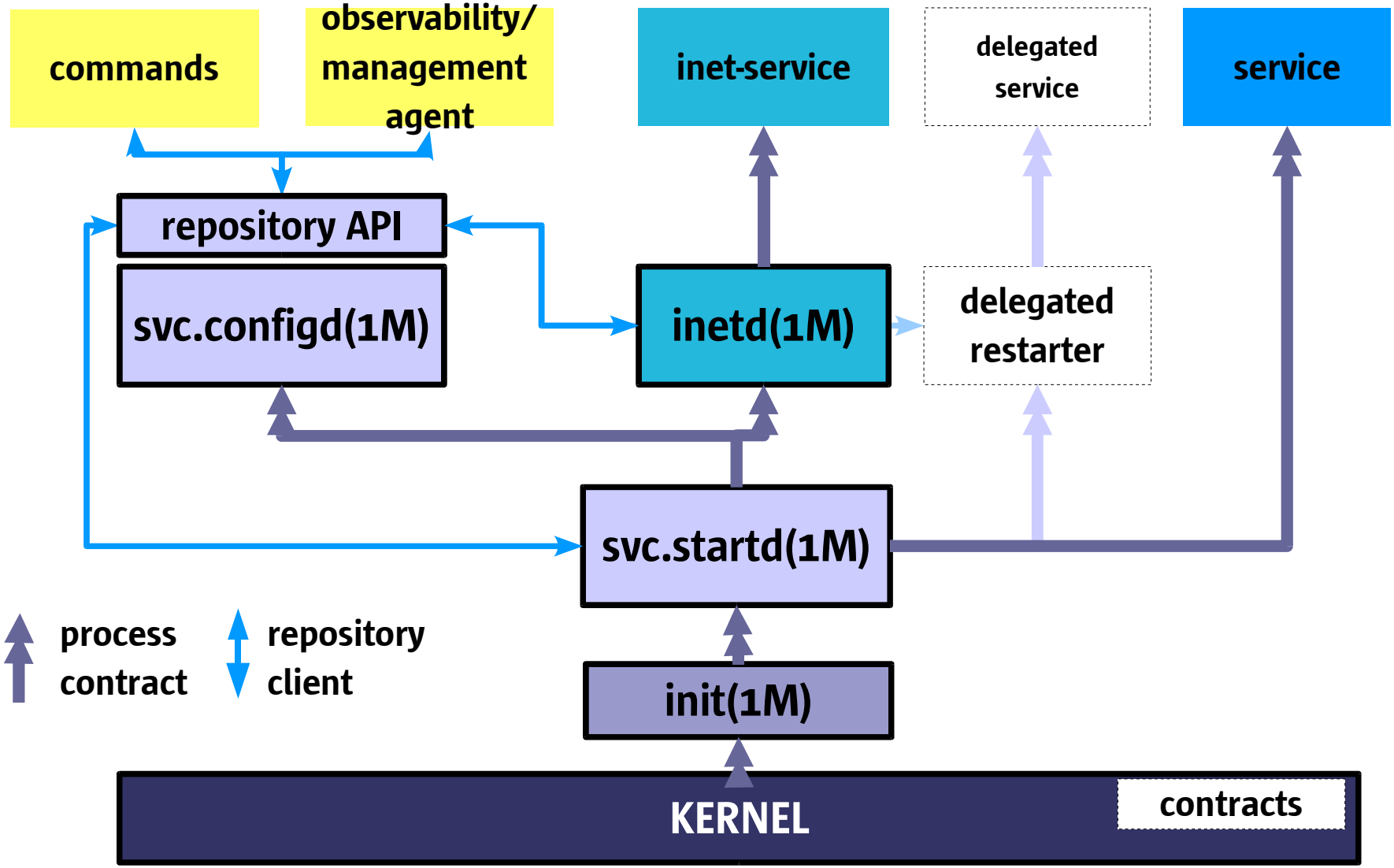
Features

- A service is now a first class object that can be managed and observed
- Legacy mechanisms still work
- Automated restart of services in dependency order:
  - administrative error
  - software bug
  - uncorrectable hardware error
- Parallel startup

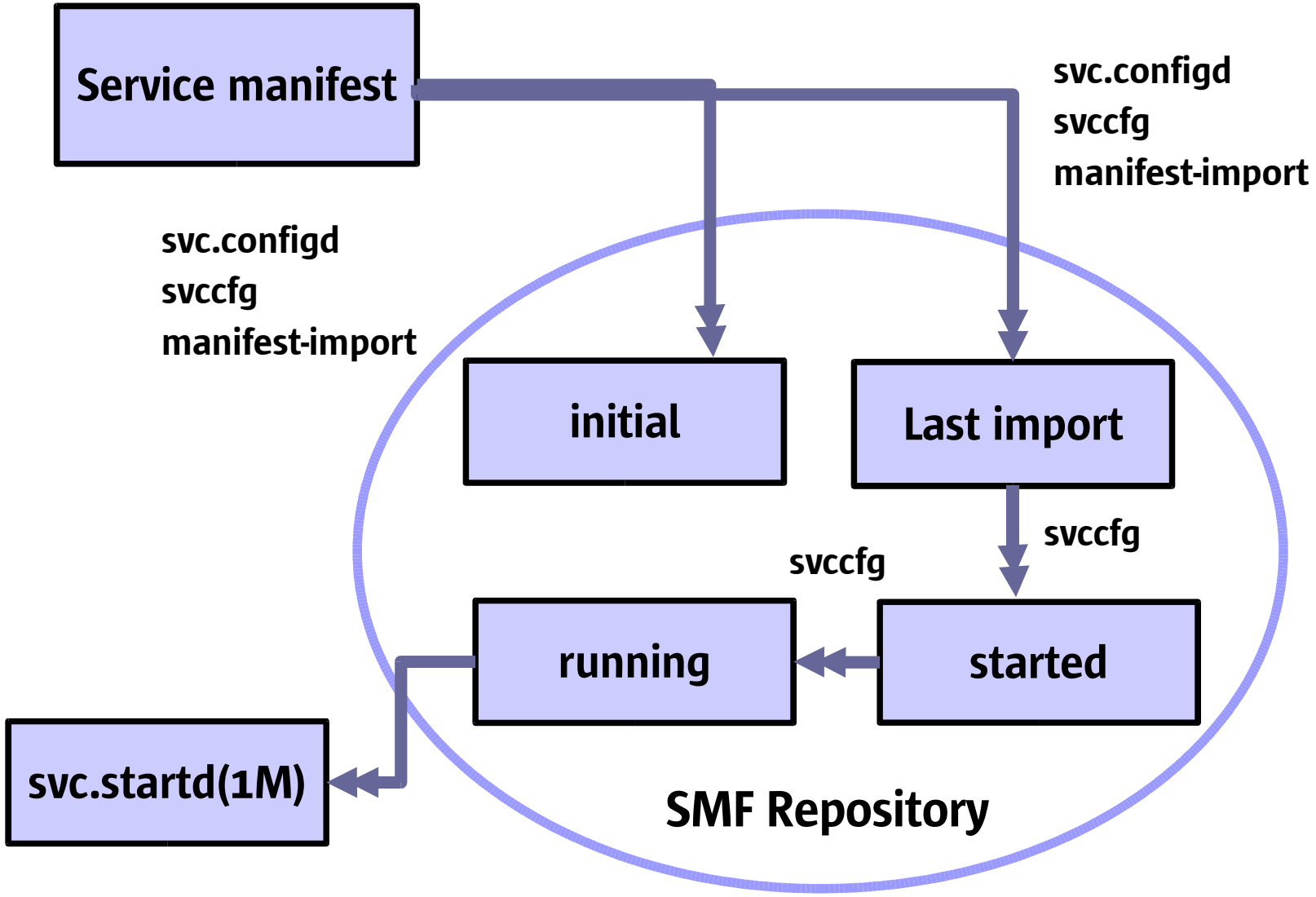
Features

- Easy access to information about misconfigured/misbehaving services
  - Easy to script!
- Admins can get a meaningful system view
- Supported enable/disable; changes persist across upgrades and patches
- Securely delegate tasks to non-root users
- Snapshots and repository backup taken automatically: restore, undo

Components: Architecture schematic



Components: Manifest and Repository



Perform MySQL setup

- See /etc/sfw/mysql/README.solaris.mysql
  - There is a typo in the installation instructions
  - Line 15 should read
    - **chmod -R 770 /var/mysql**
- Start database and test using mysqladmin

```
# /etc/sfw/mysql/mysql.server start
Starting mysqld daemon with databases from /var/mysql

# mysqladmin status
Uptime: 32  Threads: 1  Questions: 1  Slow queries: 0  Opens: 6
Flush tables: 1  Open tables: 0  Queries per second avg: 0.031
```

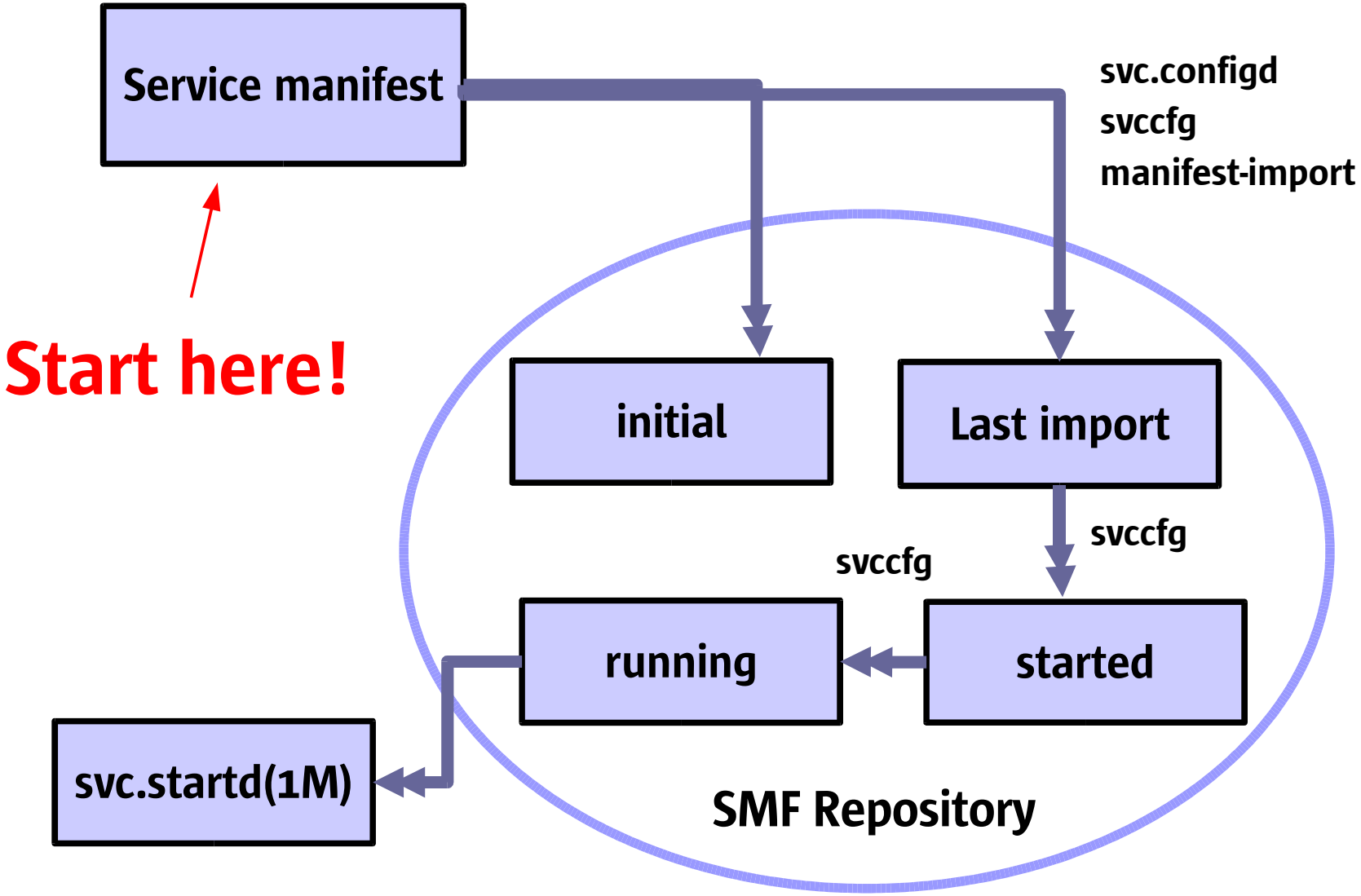
Perform MySQL setup

- Stop the database!
  - > We don't want SMF to try to start it if it is already running

```
# /etc/sfw/mysql/mysql.server stop
/etc/sfw/mysql/mysql.server stop Killing mysqld with pid 1212
Wait for mysqld to exit.050907 21:16:19  mysqld ended

 done
```

- Don't link the RC script in /etc/rc3.d

Components: Manifest and Repository



Creating an SMF Manifest

- XML file describing service properties
- Located in /var/svc/manifest/<dir>
  - site is recommended for locally developed services
- DTD at /usr/share/lib/xml/dtd/service_bundle.dtd.1
- service_bundle(4) man page

Why reinvent the wheel?

Copy and modify an existing service manifest!

Some suggested sources of manifests

- Explore `/var/svc/manifest` for similar services
  - system/utmp is a simple standalone daemon
  - system/coreadm is a simple configuration service, and
  - network/telnet is an inetd-managed daemon
- Initial inet service manifests can be created easily by invoking: `inetconv -i <file>`
- DTD is self-documenting; read it at
  `/usr/share/lib/xml/dtd/service_bundle.dtd.1`

Example: Zones Manifest

```
<service_bundle type='manifest' name='SUNWzoner:zones'>

<service  name='system/zones'  type='service' version='1'>

<create_default_instance enabled='false' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<exec_method type='method' name='start' exec='/lib/svc/method/svc-zones %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/lib/svc/method/svc-zones %m' timeout_seconds='500'> </exec_method>

<property_group name='startd' type='framework'>
            <propval name='duration' type='astring' value='transient' />
</property_group>

<stability value='Unstable' />

<template>
    <common_name>
        <loctext xml:lang='C'>  Solaris zones </loctext>
    </common_name>

    <documentation>
        <manpage title='zones' section='5' manpath='/usr/share/man' />
        <manpage title='zoneadm' section='1M' manpath='/usr/share/man' />
    </documentation>
</template>
</service>
</service_bundle>
```

Name of the service

Default state

Dependency

Start and Stop methods

Documentation

Step 1: Start with an existing manifest

- Copy a manifest that most closely resembles your new service
  - Put in /var/svc/manifest/local
- Change the following sections
  - <service name=”your service name”>
  - <create_default_instance enabled=true | false>
  - <template> to provide text descriptions of service name and man page locations.

Our new manifest – so far

```
<service_bundle type='manifest' name='MySQL'>

<service  name=”application/mysql'  type='service' version='1'>

<create_default_instance enabled='true' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<exec_method type='method' name='start' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
<exec_method type='method' name='restart' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>

<property_group name='startd' type='framework'>
            <propval name='duration' type='astring' value='transient' />
</property_group>

<stability value='Unstable' />

<template>
    <common_name>
        <loctext xml:lang='C'>  MySQL Database </loctext>
    </common_name>

    <documentation>
        <manpage title='mysql' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqld' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqladmin' section='1' manpath='/usr/sfw/man' />
    </documentation>
</template>
</service>
</service_bundle>
```

Name of the service

Default state

Documentation references

Step 2: Start and Stop methods

- You already have these – the legacy RC scripts
- Remove links from the rc<n>.d directories
- Leave them in /etc/init.d or move to some place such as /etc/opt/svc/method
- Modify start, stop, and possibly restart methods to point to the correct RC scripts
  - Use %m in exec name to pass “start”, “stop”, and “restart”
  - exec=”:kill -<signal>” is simple shortcut to replace pkills
  - Use timeout_seconds if things take a long time to complete
- See smf_method(5) for more information

Our new manifest – so far

```
<service_bundle type='manifest' name='MySQL'>

<service  name=”application/mysql'  type='service' version='1'>

<create_default_instance enabled='true' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<exec_method type='method' name='start' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
<exec_method type='method' name='restart' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>

<property_group name='startd' type='framework'>
            <propval name='duration' type='astring' value='transient' />
</property_group>

<stability value='Unstable' />

<template>
    <common_name>
        <loctext xml:lang='C'>  MySQL Database </loctext>
    </common_name>

    <documentation>
        <manpage title='mysql' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqld' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqladmin' section='1' manpath='/usr/sfw/man' />
    </documentation>
</template>
</service>
</service_bundle>
```

Start and Stop methods

Part 3: Dependencies

- This is where things begin to get complicated
- A dependency graph is invaluable
- Start with the obvious ones
  - Run level
  - Availability of file systems
  - Configuration files
- Then add dependencies on other services
  - Need to know the error boundaries – what happens to this service when other services go offline

Service Dependencies

- Dependencies can be to another SMF managed service or a file
- Types of dependencies
  - require_all – all must be online or degraded
  - require_any – at least one online or degraded
  - optional_all – all are online, disabled, degraded, or in maintenance
  - exclude_all – all are disabled, in maintenance, or not present (files)

Dependency actions

- What do to when a dependent service changes state
  - Service failure
  - Administrative stop
  - Refreshed due to property changes
- Controlled by restart_on attribute of dependency

	restart_on attribute			
Reason for Service Stop	None	Error	Restart	Refresh
Error	No	Yes	Yes	Yes
Non error stop	No	No	Yes	Yes
Refresh	No	No	No	Yes

Example of a run level dependency

Example of a Run Level 3 dependency

```
<dependency
        name='multi-user-server'
        type='service'
        grouping='require_all'
        restart_on='none'>
  <service_fmri value='svc:/milestone/multi-user-server' />
  </dependency>
```

Use svc:/milestone/multi-user for Run Level 2 services

Local filesystems dependency

```
<dependency
        name='filesystem'
        grouping='require_all'
        restart_on='none'
        type='service'>
         <service_fmri value='svc:/system/filesystem/local'/>
</dependency>
```

Be careful with this one: an error from mountall(1M) might prevent service from starting

Local filesystems dependency (cont)

Other file system services

<service_fmri value='svc:/system/filesystem/root'/>
<service_fmri value='svc:/system/filesystem/usr'/>
<service_fmri value='svc:/system/filesystem/minimal'/>

These are just services, so use SMF to find the mount scripts to see what's really happening!

```
# svcprop -p start/exec minimal
/lib/svc/method/fs-minimal
```

Dependency on a configuration file

```
<dependency
        name='database_configuration_file'
        type='path'
        grouping='require_all'
        restart_on='refresh'>
 <service_fmri value='file://localhost/var/mysql/my.cnf' />
 </dependency>
```

Extremely handy!!!! How many times have you wondered why the NFS server didn't start ??????

Config file dependency in action

- We notice that MySQL isn't starting

```
# svcs mysql
STATE          STIME    FMRI
offline        15:17:09 svc:/application/mysql:default
```

- Using SMF we can ask why

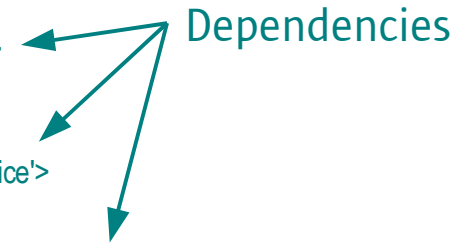
```
# svcs -l mysql
fmri         svc:/application/mysql:default
name         MySQL Database Server
enabled      true
state        offline
next_state   none
state_time   Wed Sep 07 15:17:09 2005
logfile      /var/svc/log/application-mysql:default.log
dependency   require_all/refresh file://localhost/var/mysql/my.cnf (absent)
```

MySQL dependencies

- Started at Run Level 3
- After all local file systems are mounted
- Only if there is a configuration file
  - Sample configuration files can be found in /usr/sfw/share/mysql
  - Will keep MySQL from being started in a local zone where setup process hasn't been completed
- Any others ?

Our new manifest – so far

```
<service_bundle type='manifest' name='MySQL'>

<service  name=”application/mysql'  type='service' version='1'>

<create_default_instance enabled='true' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<dependency    name='filesystem'    grouping='require_all'    restart_on='none'    type='service'>
        <service_fmri value='svc:/system/filesystem/local'/>
 </dependency>

  <dependency    name='database_configuration'    type='path'    grouping='require_all'    restart_on='refresh'>
        <service_fmri value='file://localhost/var/mysql/my.cnf' />
   </dependency>

<exec_method type='method' name='start' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
<exec_method type='method' name='restart' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
```



Transient services

- Some services should not be restarted when all processes terminate
  - > Example: one time initializations

```
<property_group name='startd' type='framework'>
                    <propval name='duration' type='astring'
   value='transient' />
</property_group>
```

- Include in SMF manifest or modify later using svcadm(1M)

Let's import our service

- Use svccfg or /lib/svc/method/manifest-import

```
# svcs mysql
svcs: Pattern 'mysql' doesn't match any instances
STATE          STIME    FMRI

# /lib/svc/method/manifest-import
Loaded 1 smf(5) service descriptions

# svcs mysql
STATE          STIME    FMRI
online         22:39:54 svc:/application/mysql:default

# mysqladmin status
Uptime: 1399  Threads: 1  Questions: 1  Slow queries: 0  Opens: 6
Flush tables: 1  Open tables: 0  Queries per second avg: 0.001
```

SMF in action

- Let's terminate the database daemon and see what happens

```
# svcs mysql
STATE          STIME    FMRI
online         22:39:54 svc:/application/mysql:default

# pkill mysqld

# svcs mysql
STATE          STIME    FMRI
online         22:45:45 svc:/application/mysql:default
```

Next steps

- Common:
  - Trivial: create simple service manifest, convert init scripts to service methods, minimal testing
  - Add privileges for delegated administration
  - Enable resource management
  - Full restartability: split monolithic services, each separately restartable component becomes its own service
- Advanced:
  - Customized error/restart handling: avoid service restart if fault can be internally handled

Troubleshooting common errors

- Start with svcs -x
- Make sure MySQL isn't running from a previous step
  - Service will fail to start and go into maintenance state
- Check the service log
  - Can be found using svcprop -p restarter/logfile mysql
- Check dependencies (svcs -l mysql)

Using SMF to diagnose common zone configuration problem

- Configured and installed local zone
- Boot the local zone
- Can zlogin to the local zone, but no network access
- First check svcs -a and look for uninitialized services
  - Telltale sign that SMF not started due to incomplete initialization
  - Finish sysidconfig
  - zlogin -C to complete identification process

Using SMF to diagnose common zone configuration problem

```
# zoneadm -z zone1 boot
# zlogin zone1
# svcs -a
...
offline         20:58:01 svc:/system/sysidtool:system
offline         20:58:01 svc:/milestone/sysconfig:default
...
uninitialized   20:58:01 svc:/network/rpc/gss:default
uninitialized   20:58:01 svc:/application/font/stfsloader:default
uninitialized   20:58:02 svc:/application/print/rfc1179:default
uninitialized   20:58:02 svc:/application/x11/xfs:default
...
```

Additional Resources

- Additional quickstart and developer documentation available at http://www.sun.com/bigadmin/content/selfheal/
- Solaris System Administration Guide http://docs.sun.com/app/docs/doc/817-1985
- Blogs:
  - http://blogs.sun.com/sch
  - http://blogs.sun.com/lianep
  - http://blogs.sun.com/jwadams
  - http://blogs.sun.com/dep
  - http://blogs.sun.com/dminer
  - http://blogs.sun.com/bobn (this workshop)

Questions ?

Thank you!

SOLARIS BOOT CAMP II
Migrating a Legacy RC service

Bob Netherton
bob.netherton@sun.com
http://blogs.sun.com/bobn