

Open Source High Availability

Writing Resource Agents for your own services

Lars Marowsky-Brée

Team Lead

SUSE Labs

imb@suse.de

Novell.[®]

Agenda

- Introduction
- Resource Agents in context

- Basic Resource Agents (+ *code*)
- Extended Resource Agents (+ *code*)

- Testing

- Questions and answers

The background of the slide is a vibrant green color with a pattern of diagonal stripes in varying shades of green, creating a sense of movement and depth. The stripes are most prominent on the right side and fade towards the left.

Context

Where do resource agents fit in?

- The goal of the cluster is to:
 - **Protect the data**
 - **To keep the services up**
 - > **fail- or switch-over**
- The cluster manages many different services.
- The operations needed to perform are quite similar.
- Resource Agents provide the “glue” between the cluster stack and the actual service:
- Provide a common, parameterized interface to manage the services

Objects: “Primitive” resources

- Class:
 - LSB, heartbeat, or **OCF Resource Agent**
- Provider
 - (for OCF only)
- Type:
 - Filesystem, Apache, Database ...
- Instance parameters:(global / per-node)
 - Supplied to the RA to identify the resource
 - Supported operations and/or special timeouts

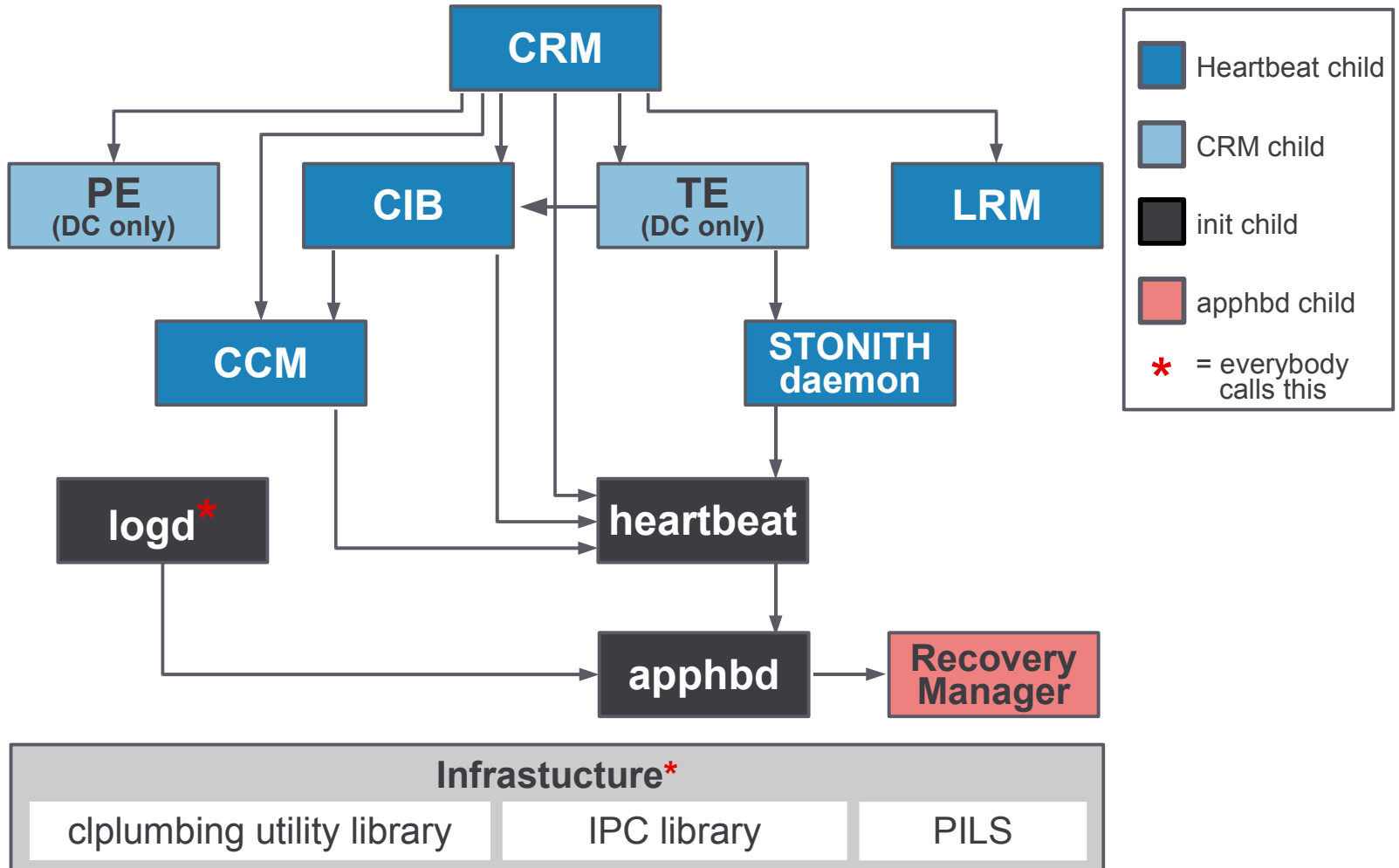
Heartbeat 2.0.x already provides ...

- OCF Resource Agents for
 - **SAP - Developed together with SAP LinuxLab**
 - **Oracle**, MySQL, IBM DB2, Postgres
 - **Xen**
 - IBM Web Application Server
 - Filesystems, Webservers, IP addresses, FTP, drbd, ...
 - Easily extended with own plugins

Major System Components

- Heartbeat master control process spawns:
 - Local Resource Manager (LRM), which spawns:
 - > Resource Agents, which manage the services
 - > STONITH Agents tie-in here
 - Cluster Consensus Membership Layer (CCM)
 - Cluster Resource Manager (CRM), which spawns:
 - > Cluster Information Base (CIB) (*r/w on the DC node only*)
 - > Policy Engine (PE) (*on the DC node only*)
 - > Transition Engine (TE) (*on the DC node only*)

Architecture



Basic Flow of Events

- Heartbeat reports node liveness event
 - Consensus Membership synchronizes across cluster
- CRM (re-)elects a new Designated Coordinator (DC)
- Elected Designated Coordinator orchestrates response to events:
 - Nodes coming and going
 - Resources reporting monitor failures
 - Administrative changes to the configuration
- Policy Engine computes needed operations
- Transitioner puts plan into action
 - LRM calls resource agents to (re-)start services
- Final cluster state: Idle.

The background of the slide is a solid green color with a pattern of diagonal stripes in a lighter shade of green, creating a sense of movement and depth.

Basic operations

OCF Resource Agent fundamentals

- Installed under `/usr/lib/ocf/resource.d/provider/Type`
 - Use your own provider name to not clash with heartbeat's own
- Configured parameters (instance_attributes):
 - Passed in environment with prefix
 - `$OCF_RESKEY_name=Value`
 - meta_attributes -> `$OCF_RESKEY_crm_meta_...`
- One commandline argument:
 - The requested operation
- Must return well defined exit codes
- Shell library provides common functions:
 - `ocf_log (crit|err|warn|info|debug) "Log message"`

Basic goals of a RA

- Provide common abstraction layer
- Start, stop, monitor the service
- Provide meta-data about the service for configuration
- Validate configuration
- Support some extended operations going forward

Deciding on good resource parameters

- **Must** clearly identify one resource instance
- Should allow several instances to be run on one node
- Should be flexible enough to customize the RA
- Should not overload the cluster configuration
- If the configuration is very complex, point to a configuration file
- IP address, for example, is small enough to carry all its configuration in the attributes

start

- After this operation has completed, service should be started
- Starting an already started service must not be an error!
- Service must be functional after start returns.

stop

- After this operation has completed, service must be fully stopped!
- Stopping an already started service must not be an error!
- Might want to implement shutdown escalation
 - If after 50% of `$OCF_RESKEY_crm_meta_timeout` the service isn't stopped, kill it by force.
- A stop failure is considered fatal:
 - The resource cannot be started anywhere else
 - The cluster must resort to extended measures to clean up!

monitor

- Return the status of the service:
- 0 (\$OCF_SUCCESS):
 - Service running fine
- 7 (\$OCF_NOT_RUNNING):
 - Service NOT running at all
- 1 (\$OCF_ERR_GENERIC):
 - Considered **active** but failed
- This three-way distinction is **NOT** optional!

Startup monitor call (probes)

- CRM checking for active resources during startup
 - Why? Failures and administrative accidents.
- Called **everywhere, on all nodes**
- Be careful to not return errors for resources which are stopped, otherwise recovery will begin

meta-data

- Print an XML chunk to stdout, describing
 - The RA itself
 - The parameters it accepts
 - > Name
 - > Description
 - > Type (Integer, string)
 - > Suggested default
 - Operations supported
 - > Name
 - > Suggested timeouts and repeat intervals
- Descriptions can be localized

meta-data

```
meta_data() {
    cat <<END
    <?xml version="1.0"?>
    <!DOCTYPE resource-agent SYSTEM "ra-api-1.dtd">
    <resource-agent name="ClusterMon">
    <version>1.0</version>

    <longdesc lang="en">
    This is a ClusterMon Resource Agent.
    It outputs current cluster status to the html.
    </longdesc>
    <shortdesc lang="en">ClusterMon resource agent</shortdesc>
```

meta-data (parameters)

```
<parameters>
<parameter name="user" unique="0">
  <longdesc lang="en">The user we want to run crm_mon as</longdesc>
  <shortdesc lang="en">The user we want to run crm_mon as</shortdesc>
  <content type="string" default="root" />
</parameter>
...
</parameters>
```

validate-all

- Validate the instance attributes
- Syntax checking
- Be careful:
 - This is called independently from start/stop state
 - Not all prerequisites may be present
- Not actually called by anyone right now ;-)

The background of the slide is a solid green color with a pattern of diagonal stripes in varying shades of green, creating a sense of movement and depth. The stripes are most prominent on the right side and fade towards the left.

Advanced Resources

Beyond start/stop

- Wouldn't it be nice if the cluster could move resources without shutting them down?
- ***meta_attribute: allow_migrate***
- If the source and the target are healthy,
- *migrate_to* will be invoked to *push* the resource to
 - *OCF_RESKEY_CRM_meta_migrate_target*
- *migrate_from* will be invoked to *pull* the resource and complete the migration
- If either one fails, reverts to stop/start
- Used to live migrate Xen guests.

What are Clones?

- A convenient way to start N copies of a resource on M nodes
- Types: Anonymous, Globally Unique
- Two extra environment variables:
 - OCF_RESKEY_crm_meta_clone,
 - OCF_RESKEY_crm_meta_clone_max
- Currently used to support OCFS2 and fencing
- Must use an OCF type resource agent
- See <http://www.linux-ha.org/v2/Concepts/Clones>

Probes & Clones

- The CRM will send a probe for each clone instance that may be active on the machine
- Anonymous Clones:
 - Nodes must respond with `OCF_SUCCESS` to **exactly N of these probes**.
 - N is the number of active clone instances on the node
 - All other probes: `OCF_NOT_RUNNING`

Probes & Clones (cont'd)

- The CRM will send a probe for each clone instance that may be active on the machine
- Globally Unique Clones:
 - Nodes must respond with `OCF_SUCCESS` **only if the particular clone instance is active.**
 - Check `OCF_RESKEY_clone` and `OCF_RESKEY_clone_max` to be sure
 - All other probes: `OCF_NOT_RUNNING`

Clone: receiving peer information

- Optional extra operation: **notify**
 - meta_attribute: “*notify*” set to “1”
- Two types of notifications: *pre* & *post*
- Extra environment variables:
 - OCF_RESKEY_crm_meta_globally_unique (default: true)
 - OCF_RESKEY_crm_meta_notify_type
 - OCF_RESKEY_crm_meta_notify_operation
 - OCF_RESKEY_crm_meta_notify_{x}_resource
 - OCF_RESKEY_crm_meta_notify_{x}_uname
 - {x} : start, stop, active, inactive

Multi-state resources

- Superset of a clone
 - Can be either anonymous or unique
- Currently supports two states:
 - After “start” always in “slave” mode
 - promotion/demote are separate operations
- Used mostly for active/passive replication scenarios (drbd)

Multi-State RA Extensions

- Operations
 - **promote**, **demote**, *optional: notify*
- Additional *monitor* return values
 - \$OCF_RUNNING_MASTER
 - \$OCF_FAILED_MASTER
- Extra environment variables:
 - OCF_RESKEY_crm_meta_notify_{x}_resource
 - OCF_RESKEY_crm_meta_notify_{x}_uname
 - {x} : promote, demote, master, slave

Multi-state: defining master ability

- During start or post-notify for start:
 - `crm_master -v X` **must** be called
 - $X = 0$ (or not called): cannot become master
 - $X > 0$: Highest value node will be chosen as master
- *monitor* can adjust the values, if needed
- Example: drbd resource agent implements this

The background of the slide is a solid green color with a pattern of diagonal stripes in varying shades of green, creating a sense of movement and depth. The stripes are oriented from the top-left to the bottom-right.

Testing ...

Setup the environment

- Make sure all dependencies are started locally
- Setup the environment variables
- Call the agent manually with the desired operations
- Or use our automated test harness

Testing Your Agent

```
# /usr/sbin/ocf-tester -n Test -o ip=172.16.1.1 -o  
netmask=255.255.255.0 /usr/lib/ocf/resource.d/heartbeat/IPaddr
```

Beginning tests for /usr/lib/ocf/resource.d/heartbeat/IPaddr...

- * Your agent does not support the notify action (optional)
- * Your agent does not support the demote action (optional)
- * Your agent does not support the promote action (optional)
- * Your agent does not support master/slave (optional)
- * Your agent does not support the reload action (optional)

/usr/lib/ocf/resource.d/heartbeat/IPaddr **passed all tests**

The background of the slide is a solid green color with a pattern of diagonal stripes in varying shades of green, creating a sense of movement and depth. The stripes are most prominent on the right side and fade towards the left.

Curious...?

Where to go From Here:

See also IO125, TUT234

Test using Xen!

Build your highly available media archive at home using openSUSE™

Book recommendations:

Blueprints for High Availability by Evan Marcus and Hal Stern
The Linux Enterprise Cluster by Karl Kopper



Online resources:

<http://www.linux-ha.org/>
<http://www.lcic.org/>



Questions? Feedback?

Novell®

Unpublished Work of Novell, Inc. All Rights Reserved.

This work is an unpublished work and contains confidential, proprietary, and trade secret information of Novell, Inc. Access to this work is restricted to Novell employees who have a need to know to perform tasks within the scope of their assignments. No part of this work may be practiced, performed, copied, distributed, revised, modified, translated, abridged, condensed, expanded, collected, or adapted without the prior written consent of Novell, Inc. Any use or exploitation of this work without authorization could subject the perpetrator to criminal and civil liability.

General Disclaimer

This document is not to be construed as a promise by any participating company to develop, deliver, or market a product. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. Novell, Inc. makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The development, release, and timing of features or functionality described for Novell products remains at the sole discretion of Novell. Further, Novell, Inc. reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All Novell marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States and other countries. All third-party trademarks are the property of their respective owners.

