A Self-Adaptive Deployment Framework for Service-Oriented Systems

Sander van der Burg  Eelco Dolstra

Delft University of Technology, EEMCS, Department of Software Technology

May 24, 2011
The Service Oriented Computing (SOC) paradigm is very popular to build distributed applications.
Systems are composed of distributable components (or services), working together to achieve a common goal.
Services are deployed in *networks* of machines having various properties and capabilities.
Various events may occur in a network of machines:

- Crashing machines
- Adding a new machine
- Change of a capability (e.g. increase of RAM)
Implications

Events have a number of *implications*:

- The system may partially or completely *break*
- The deployment may become *suboptimal*
In case of an event, a redeployment may be performed.

(Re)deployment is difficult, time consuming and error prone.
Software deployment

All of the activities that make a software system available for use

Carzaninga et al.
Some deployment tools *reconfigure* service behaviour and connections, but do not redeploy a complete system.

A number of deployment tools are designed for *specific* component technologies:
- e.g. DRACO, OSGi

A number of *deployment planning* algorithms have been developed, but little of them are used in tooling which perform *actual* deployment.
Disnix: A distributed service deployment tool

$ disnix-env -s services.nix -i infrastructure.nix -d distribution.nix

A Self-Adaptive Deployment Framework
Disnix: A Distributed Service Deployment tool

Disnix is built on top of the *purely functional* deployment properties of the Nix package manager:

- A *generic* approach; Many types of services can be supported
- Declarative language to describe components and infrastructure
- Store multiple versions/variants of components safely next to each other
- (Almost) atomic upgrades/rollbacks
- Garbage collector to safely remove obsolete components
How to *extend* Disnix to deal with events occurring in a network?
Dynamic deployment extension

A Self-Adaptive Deployment Framework

infrastructure generator

infrastructure.nix

infrastructure augmenter

distribution generator

services.nix

qos.nix

augment.nix

disnix-env

User input

infrastructure.nix
Dynamic deployment extension

The infrastructure generator *discovers* machines in the network and their properties.

The infrastructure augmenter *adds* privacy sensitive properties to the infrastructure model.

The QoS model *maps* services to machines based on a user-defined QoS policy.

The generated infrastructure and distribution model are used by Disnix to perform *redeployment*. 
The distribution generator uses a QoS model to implement a distribution policy.

Initially, all services are mapped to all machines in the network.

By composing functions, a desirable distribution can be generated.
Typically, a QoS model has the following structure:

- **Candidate host selection phase.** For each service, decide which host is suitable by looking to service and system properties.
- **Division phase.** Divide services across the candidate hosts, using some strategy.
To create a candidate host selection, attributes defined in a services model must be *mapped* onto services defined in the infrastructure model:

```json
Service model
{
    ZipcodeService = {
        name = "ZipcodeService";
        type = "tomcat-webapplication";
        zone = "US";
    },

    RoomService = {
        name = "RoomService";
        type = "tomcat-webapplication";
        zone = "Europe";
    }
}

Infrastructure model
{
    test1 = {
        hostname = "test1.example.org";
        zone = "Europe";
        supportedTypes = [ "tomcat-webapplication" "process" ];
    },

    test2 = {
        hostname = "test2.example.org";
        zone = "US";
        supportedTypes = [ "tomcat-webapplication" ];
    }
}
```
Various division functions are supported:
- One-dimensional (highest bidder, lowest bidder, greedy)
- Multi-way cut approximation
  - Reduce the amount of network links
  - A. Heydarnoori et al
- Minimum set cover approximation
  - Cost efficient deployment
  - A. Heydarnoori et al
- Vertex coloring approximation
  - Maximize the number of network links using the minimum amount of machines

Framework is *extensible*; custom filter functions can be easily implemented
Example cases:

- **Staff Tracker.** Toy SOA system which comes in two variants (PHP/MySQL and Webservices).
- **ViewVC.** Open-source web based Subversion repository viewer ([http://viewvc.tigris.org](http://viewvc.tigris.org)).
- **SDS2.** Industrial SOA case study from Philips Research.
Evaluation method:

- Each case study is tested in a network of 20 Xen Virtual machines.
- For each case study, we fire various events: adding machines, crashing machines.
- We measure the times of the redeployment phases for each event.
### Evaluation

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Discovery (s)</th>
<th>Generation (s)</th>
<th>Build (s)</th>
<th>Distribution (s)</th>
<th>Transition (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial deployment</td>
<td>1.0</td>
<td>27.9</td>
<td>31.9</td>
<td>1.2</td>
<td>1.6</td>
</tr>
<tr>
<td>1</td>
<td>Add MySQL server 2</td>
<td>1.0</td>
<td>8.3</td>
<td>4.7</td>
<td>1.7</td>
<td>1.7</td>
</tr>
<tr>
<td>2</td>
<td>Add MySQL server 3</td>
<td>1.0</td>
<td>4.4</td>
<td>4.5</td>
<td>1.2</td>
<td>1.3</td>
</tr>
<tr>
<td>3</td>
<td>Add Apache server 2</td>
<td>1.0</td>
<td>4.4</td>
<td>4.4</td>
<td>1.3</td>
<td>1.5</td>
</tr>
<tr>
<td>4</td>
<td>Remove MySQL server 2</td>
<td>1.0</td>
<td>8.2</td>
<td>4.6</td>
<td>1.5</td>
<td>1.8</td>
</tr>
<tr>
<td>5</td>
<td>Remove Apache server 2</td>
<td>1.0</td>
<td>8.8</td>
<td>4.4</td>
<td>0.9</td>
<td>1.2</td>
</tr>
</tbody>
</table>

### StaffTracker (Web services)

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Discovery (s)</th>
<th>Generation (s)</th>
<th>Build (s)</th>
<th>Distribution (s)</th>
<th>Transition (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial deployment</td>
<td>1.0</td>
<td>28.6</td>
<td>169.8</td>
<td>13.1</td>
<td>4.0</td>
</tr>
<tr>
<td>1</td>
<td>Add MySQL server $500</td>
<td>1.0</td>
<td>10.7</td>
<td>17.7</td>
<td>6.1</td>
<td>2.9</td>
</tr>
<tr>
<td>2</td>
<td>Add Tomcat server $500</td>
<td>1.0</td>
<td>8.4</td>
<td>10.2</td>
<td>13.7</td>
<td>5.7</td>
</tr>
<tr>
<td>3</td>
<td>Remove Tomcat server $500</td>
<td>1.0</td>
<td>4.2</td>
<td>0.5</td>
<td>0.6</td>
<td>2.0</td>
</tr>
<tr>
<td>4</td>
<td>Remove MySQL server $500</td>
<td>1.0</td>
<td>4.2</td>
<td>0.5</td>
<td>1.2</td>
<td>1.8</td>
</tr>
</tbody>
</table>

### ViewVC

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Discovery (s)</th>
<th>Generation (s)</th>
<th>Build (s)</th>
<th>Distribution (s)</th>
<th>Transition (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial deployment</td>
<td>1.0</td>
<td>28.5</td>
<td>555.9</td>
<td>9.5</td>
<td>17.5</td>
</tr>
<tr>
<td>1</td>
<td>Add MySQL server</td>
<td>1.0</td>
<td>8.2</td>
<td>0.6</td>
<td>0.6</td>
<td>1.1</td>
</tr>
<tr>
<td>2</td>
<td>Add Apache server</td>
<td>1.0</td>
<td>8.4</td>
<td>0.5</td>
<td>0.6</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>Add Subversion server + repo</td>
<td>1.0</td>
<td>8.4</td>
<td>6.3</td>
<td>3.7</td>
<td>8.2</td>
</tr>
<tr>
<td>4</td>
<td>Remove Apache server 1</td>
<td>1.0</td>
<td>8.4</td>
<td>4.6</td>
<td>7.5</td>
<td>1.9</td>
</tr>
<tr>
<td>5</td>
<td>Remove MySQL server 1</td>
<td>1.0</td>
<td>8.3</td>
<td>5.7</td>
<td>1.9</td>
<td>1.8</td>
</tr>
</tbody>
</table>

### SDS2

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Discovery (s)</th>
<th>Generation (s)</th>
<th>Build (s)</th>
<th>Distribution (s)</th>
<th>Transition (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial deployment</td>
<td>1.0</td>
<td>28.6</td>
<td>642.9</td>
<td>122.8</td>
<td>147.7</td>
</tr>
<tr>
<td>1</td>
<td>Add MySQL server 2 (hospital)</td>
<td>1.0</td>
<td>6.7</td>
<td>0.6</td>
<td>19.1</td>
<td>1.9</td>
</tr>
<tr>
<td>2</td>
<td>Add Tomcat server 2 (Philips)</td>
<td>1.0</td>
<td>8.4</td>
<td>0.6</td>
<td>18.5</td>
<td>1.8</td>
</tr>
<tr>
<td>3</td>
<td>Remove Tomcat server 1 (Philips)</td>
<td>1.0</td>
<td>8.3</td>
<td>360.2</td>
<td>26.9</td>
<td>6.3</td>
</tr>
<tr>
<td>4</td>
<td>Remove MySQL server 1 (hospital)</td>
<td>1.2</td>
<td>10.5</td>
<td>370.4</td>
<td>120.8</td>
<td>124.3</td>
</tr>
</tbody>
</table>

Sander van der Burg, Eelco Dolstra  A Self-Adaptive Deployment Framework
## Evaluation

### StaffTracker (PHP/MySQL)

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Discovery (s)</th>
<th>Generation (s)</th>
<th>Build (s)</th>
<th>Distribution (s)</th>
<th>Transition (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial deployment</td>
<td>1.0</td>
<td>27.9</td>
<td>31.9</td>
<td>1.2</td>
<td>1.6</td>
</tr>
<tr>
<td>1</td>
<td>Add MySQL server 2</td>
<td>1.0</td>
<td>8.3</td>
<td>4.7</td>
<td>1.7</td>
<td>1.7</td>
</tr>
<tr>
<td>2</td>
<td>Add MySQL server 3</td>
<td>1.0</td>
<td>4.4</td>
<td>4.5</td>
<td>1.2</td>
<td>1.33</td>
</tr>
<tr>
<td>3</td>
<td>Add Apache server 2</td>
<td>1.0</td>
<td>4.4</td>
<td>4.4</td>
<td>1.3</td>
<td>1.54</td>
</tr>
<tr>
<td>4</td>
<td>Remove MySQL server 2</td>
<td>1.0</td>
<td>8.2</td>
<td>4.6</td>
<td>1.5</td>
<td>1.85</td>
</tr>
<tr>
<td>5</td>
<td>Remove Apache server 1</td>
<td>1.0</td>
<td>8.8</td>
<td>4.4</td>
<td>0.9</td>
<td>1.2</td>
</tr>
</tbody>
</table>

### ViewVC

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Discovery (s)</th>
<th>Generation (s)</th>
<th>Build (s)</th>
<th>Distribution (s)</th>
<th>Transition (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial deployment</td>
<td>1.0</td>
<td>28.6</td>
<td>169.8</td>
<td>13.1</td>
<td>4.01</td>
</tr>
<tr>
<td>1</td>
<td>Add MySQL server $500</td>
<td>1.0</td>
<td>10.7</td>
<td>17.7</td>
<td>6.1</td>
<td>2.92</td>
</tr>
<tr>
<td>2</td>
<td>Add Tomcat server $500</td>
<td>1.0</td>
<td>8.4</td>
<td>10.2</td>
<td>13.7</td>
<td>5.73</td>
</tr>
<tr>
<td>3</td>
<td>Remove Tomcat server $500</td>
<td>1.0</td>
<td>4.2</td>
<td>0.5</td>
<td>0.6</td>
<td>2.04</td>
</tr>
<tr>
<td>4</td>
<td>Remove MySQL server $500</td>
<td>1.0</td>
<td>4.2</td>
<td>0.5</td>
<td>1.2</td>
<td>1.8</td>
</tr>
</tbody>
</table>

### SDS2

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Discovery (s)</th>
<th>Generation (s)</th>
<th>Build (s)</th>
<th>Distribution (s)</th>
<th>Transition (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial deployment</td>
<td>1.0</td>
<td>28.6</td>
<td>642.9</td>
<td>122.8</td>
<td>147.71</td>
</tr>
<tr>
<td>1</td>
<td>Add MySQL server 2 (hospital)</td>
<td>1.0</td>
<td>6.7</td>
<td>0.6</td>
<td>19.1</td>
<td>1.92</td>
</tr>
<tr>
<td>2</td>
<td>Add Tomcat server 2 (Philips)</td>
<td>1.0</td>
<td>8.4</td>
<td>0.6</td>
<td>18.5</td>
<td>1.83</td>
</tr>
<tr>
<td>3</td>
<td>Remove Tomcat server 1 (Philips)</td>
<td>1.0</td>
<td>8.3</td>
<td>360.2</td>
<td>26.9</td>
<td>6.3</td>
</tr>
<tr>
<td>4</td>
<td>Remove MySQL server 1 (hospital)</td>
<td>1.2</td>
<td>10.5</td>
<td>370.4</td>
<td>120.8</td>
<td>124.3</td>
</tr>
</tbody>
</table>

### Results

- **Initial deployment times are long; redeployment times are significantly shorter**
- **If a new machine is added: only a short downtime occurs when activating the new configuration**
- **In case of a crashing machine: if a network has been deployed before, you can switch instantly back**
- **In some cases SDS2 has long redeployment times, due to bad design decisions**
We have shown a self-adaptive deployment extension around Disnix.

A system gets redeployed in case of an event.

The approach is generic.

A distribution policy can be implemented, by using a collection of algorithms.

Due to the purely functional properties of Nix, redeployment is reliable and efficient.
Future work

- Support complex network topologies.
- Implement more sophisticated deployment planning algorithms:
  - e.g. Avala (*Sam Malek et al.*)
Support mutable state *generically*. Currently, the only way to deal with state:

- *Mark* services as stateful and not relocating them
- Use an infrastructure which *replicates* data automatically
Disnix (and other related Nix tooling) is available under Free and Open Source licenses:

- Disnix homepage: http://nixos.org/disnix
- Other Nix related software: http://nixos.org