**Plush: An Infrastructure for Managing and Visualizing Distributed Applications**

Jeannie Albrecht, Ryan Braud, Darren Dao, John Jersin, Christopher Tuttle, Alex C. Snoeren, and Amin Vahdat
University of California, San Diego

(jalbrecht, rbraud, cuttle, snoeren, vahdat) @ cs.ucsd.edu, (hdao, jjersin) @ ucsd.edu  http://plush.ucsd.edu

**Goal:** Provide an extensible application management system for large-scale distributed systems

**Motivation**
- **Problem:** How do we deploy, manage, and maintain distributed applications that simultaneously run on hundreds of heterogeneous physical machines?
- Existing approaches for finding resources and managing applications are cumbersome, manual, and error-prone.
- Tools exist to address some issues, but utility is limited by lack of integration.
- **Plush** provides a unified environment to support the distributed application design and deployment lifecycle on PlanetLab and in clusters.

**Architecture**
- **Plush consists of an application controller** that communicates with client processes running on each of the available resources.
- Application description specifies resources, software, program execution, synchronization requirements, and process monitoring details.

**The Plush Controller**
- Takes application description and pool of resources (optional) as input.
- Uses a resource matcher to select/allocate resources based on user criteria.
- Installs a set of user-defined software packages and application files.
- Configures and starts processes, monitoring the running application and resources throughout execution.
- Performs cleanup actions at the clients after execution completes.

**Application Building Blocks**
- Plush application descriptions are comprised of different types of blocks: `application_blocks`, `component_blocks`, `process_blocks`, and `barrier_blocks`.
- This simple file distribution application consists of two groups of resources: source and receiving clients.
- Servers prepare files for transfer, join the overlay network, and wait for clients to join the overlay before receiving files.
- Clients join the overlay and wait for servers to prepare files and join the overlay before receiving files.
- Barriers synchronize processes and components.

**Application Description**
- Plush XML document that describes the distributed application.
- Users generate the XML manually or using a GUI.

**Using the Plush Terminal**
- Plush users can interact with Plush via a shell-like terminal interface.
- The table below shows some basic Plush terminal commands.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>load &lt;filename&gt;</td>
<td>Read an XML app description</td>
</tr>
<tr>
<td>connect &lt;hostname&gt;</td>
<td>Start and connect to a Plush client on a remote host</td>
</tr>
<tr>
<td>disconnect</td>
<td>Close all open client connections</td>
</tr>
<tr>
<td>info control</td>
<td>Print the controller’s state information</td>
</tr>
<tr>
<td>run</td>
<td>Start executing the application</td>
</tr>
<tr>
<td>shell &lt;quoted string&gt;</td>
<td>Run “quoted string” as a shell command on all hosts</td>
</tr>
</tbody>
</table>

**Visualizing Plush Applications on PlanetLab**
- Nebula is a GUI for running applications with Plush.
- Users can visualize Plush applications running on PlanetLab in real time.
- Colored dots on the map below indicate available sites (grey), and running sites (red).

**Research Contributions**
- A high-level specification language for distributed computations that captures the requirements of a broad range of applications.
- Extensibility to support a range of mechanisms for resource discovery and creation, resource acquisition, software installation, and application control.
- A unified framework for distributed application deployment, deployment, and visualization.