A Reference Architecture for Distributed Software Deployment

Sander van der Burg

Delft University of Technology, EEMCS, Department of Software Technology

June 3, 2013
A Reference Architecture for Distributed Software Deployment

Sander van der Burg
A Reference Architecture for Distributed Software Deployment

Sander van der Burg
Software deployment
Software deployment

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

Software deployment

- Time consuming
- Error prone
- Destructive upgrades
Some history: Early history
Some history: High-level languages and operating systems

```c
/* Return the parsed main chunk */
return chunk;
}

IFF_Chunk *IFF_read(const char *filename, const IFF_Extension *extension, const unsigned int extensionLength)
{
    IFF_Chunk *chunk;
    FILE *file = fopen(filename, "rb");
    /* Open the IFF file */
    if(file == NULL)
    {
        IFF_error("ERROR: cannot open file: %s\n", filename);
        return NULL;
    }
    /* Parse the main chunk */
    chunk = IFF_readFd(file, extension, extensionLength);
    /* Close the file */
    fclose(file);
    /* Return the chunk */
    return chunk;
}

int IFF_writeFd(FILE *file, const IFF_Chunk *chunk, const IFF_Extension *extension, const unsigned int extensionLength)
{
    if(IFF_writeChunk(file, chunk, NULL, extension, extensionLength))
        return TRUE;
    else
        return FALSE;
}

int IFF_write(const char *filename, const IFF_Chunk *chunk, const IFF_Extension *extension, const unsigned int extensionLength)
{
    int status = TRUE;
    FILE *file = fopen(filename, "wb");
    if(file == NULL)
    {
        IFF_error("ERROR: cannot open file: %s\n", filename);
        return FALSE;
    }
```
Some history: High-level languages and operating systems

Software components

Requires compiler or interpreter and a compatible operating system
Some history: Component-based software engineering

NixOS

The Nix project

Welcome to the homepage of the Nix project, which consists of a number of related open source subprojects:

- **Nix** is a purely functional package manager. This means that it can ensure that an upgrade to one package cannot break others, that you can always roll back to previous version, that multiple versions of a package can coexist on the same system, and much more.
- **Nixpkgs** is a large collection of packages that can be installed with the Nix package manager.
- **NixOS** is a Nix-based Linux distribution. Thanks to Nix, it supports atomic upgrades, rollbacks and multi-user package management, and it has a declarative approach to system configuration management that makes it easy to reproduce a configuration on another machine.
- **Hydra** is a Nix-based continuous build system.
- **Disnix** is a Nix-based distributed service deployment system.

Latest News

PatchELF 0.6 released 2011/11/7

PatchELF 0.6 has been released. Apart from some bug fixes, it adds support for executables produced by the Gold linker, see the README for details.
Software components

- Components increase programmer productivity
- Components increase quality of software
Some history: Component-based software engineering

Disadvantages:
Nowadays: Services on the Internet
Nowadays: Services on the internet

Challenges:
Nowadays: Services on the Internet

Challenges:

Software components

Software deployment has become increasingly more complicated
Earlier research: Nix deployment system

NixOS

GNU GRUB version 0.97 (636K lower / 129984K upper memory)

Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, or 'c' to edit the
commands before booting, or 'q' for a command-line.

GNU/Linux

Sander van der Burg
A Reference Architecture for Distributed Software Deployment
Earlier research: Nix deployment system

NixOS

Nix deployment

- Fully automated deployment
- Reliable
- Reproducible
- Generic
- Efficient
Deploying service-oriented systems

Nix and NixOS are not sufficient for deploying service-oriented systems:
Nix and NixOS are not sufficient for deploying service-oriented systems:
Nix and NixOS are not sufficient for deploying service-oriented systems:

**Non-functional requirements**
- Is privacy-sensitive data secured?
- Do the analysis components perform well?
- Is the system resilient to machine crashes?
A Reference Architecture for Distributed Software Deployment

Shop Tools
An Architecture for Distributed Software Deployment

Sander van der Burg

A Reference Architecture for Distributed Software Deployment
Questions