SUSE® Build Story

An Epic Novel with Multiple Endings

Andreas Jaeger
Product Manager
SUSE

Craig Gardner
Software Engineering Manager
SUSE

Build Story?

- Deployment options
- Tools to help with deployment (“Build Story”)
- Imaging

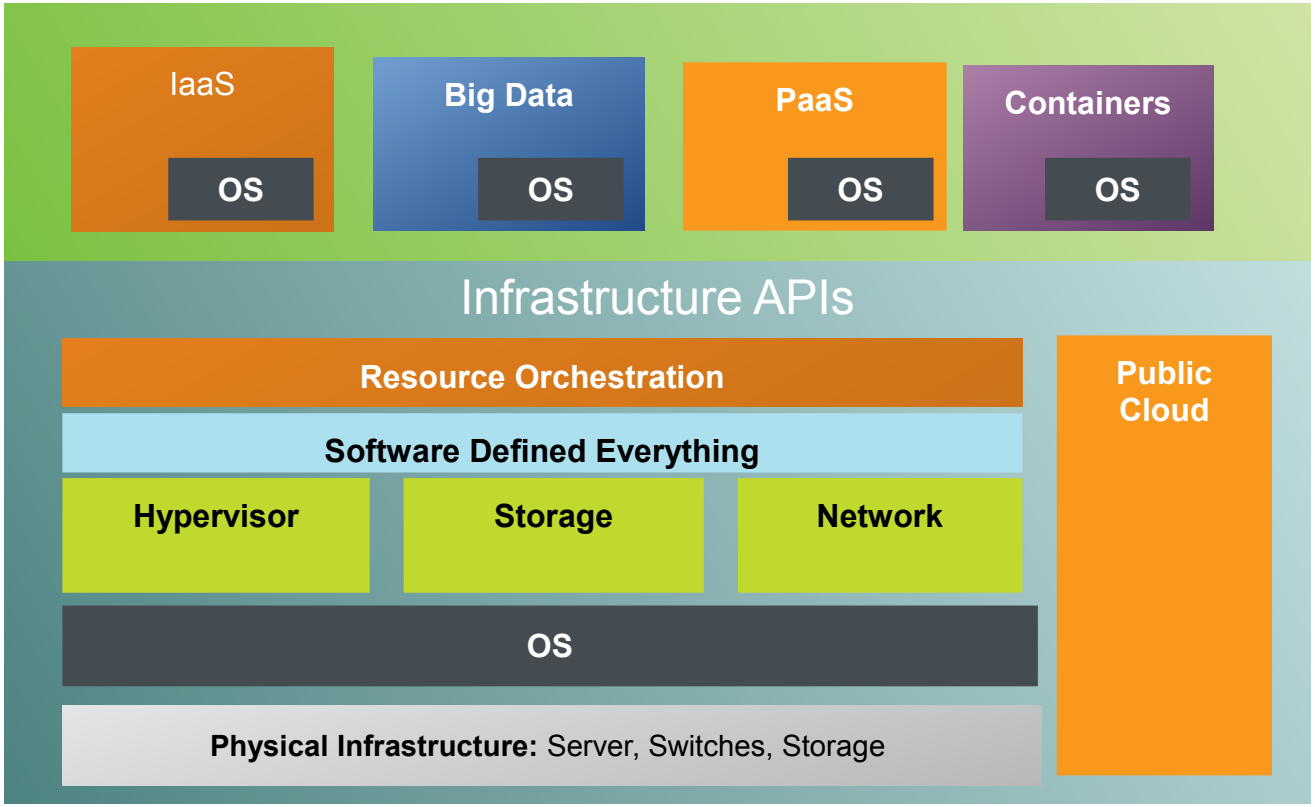
Deployment

Services

- Nobody *wants* to manage systems, configurations, etc
- User want to deploy and operate *services*
- Service delivery might include:
  - running one or more servers
  - instances of one ore more software applications

Emerging Infrastructure

Management
- **Monitoring**
- **Patching**
- **Image Creation**
- **Configuration Management**
- **Orchestration**



Deployment “targets”

- Physical server
- Virtual server – running different hypervisor
- Public cloud – like Amazon EC2, Microsoft Azure, Google Compute Engine
- Private cloud – OpenStack based with different hypervisors
- Platform-as-a-Service – examples Cloud Foundry, Heroku, OpenShift
- Container – Docker

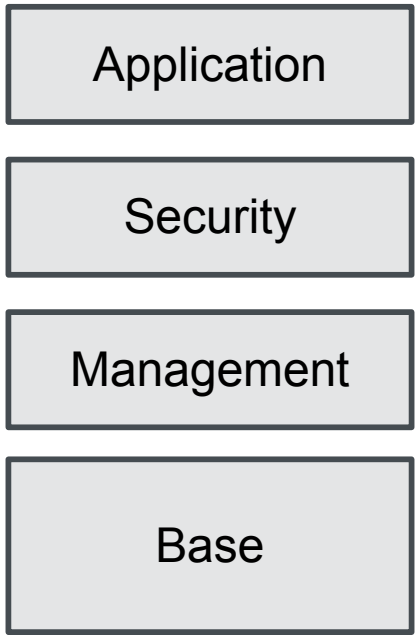
Deployment options

- Depending on Target:
  - Manual installation
  - Automatic installation (AutoYaST)
  - Imaging
  - Setup via config file, examples: Dockerfile, Application Manifest (Cloud Foundry)
  - Setup using Configuration Management Software (CMS) like Ansible, CFEngine, Chef, Puppet, and SaltStack

Deployment scope

- Deployment of full environment incl. Kernel
- Deployment on top of some base:
  - Docker
  - JeOS
  - Customized minimal Golden Image
  - PaaS (Cloud Foundry)

“Lego blocks”



- Developer wants to create workload based on top of standard components.
- IT team wants to create standard components for reuse.
- Developer wants to easily update workload after IT team updates standard components.

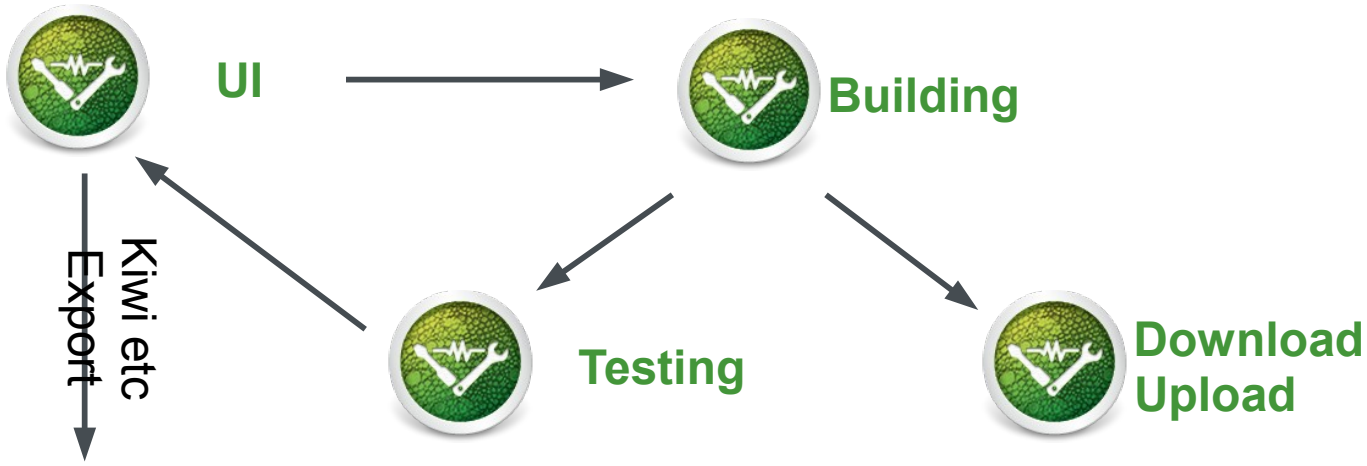
Where's the future for workloads?

Where's the future for workloads?

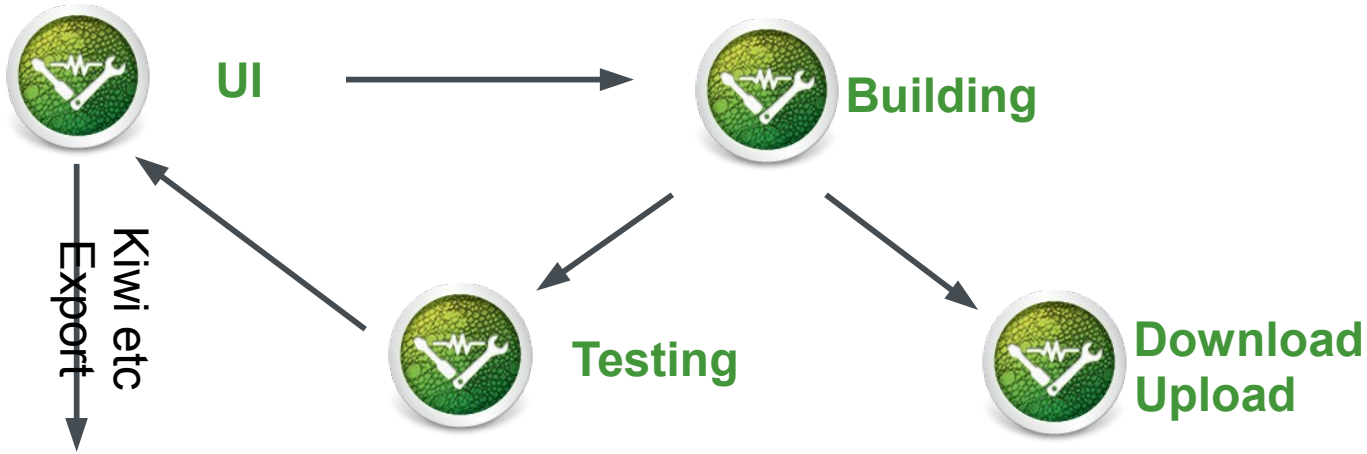
- Physical installation?
- Virtual machines?
- Cloud deployment?
- Platform-as-a-Service like Heroku, Cloud Foundry, OpenShift?
- Container (Docker)?

Architecture

Architecture “Online”



Architecture “Modular Onsite”



Architecture “Minimal Onsite”

Build Story

Build Story?

- High level use cases:
  - Developer wants to create workload based on top of standard components.
  - IT team wants to create standard components for reuse.

Infrastructure Artifacts (1/2)

- Physical server:
  - Image to run
  - AutoYaST profile
- Virtual server:
  - Image to run (Hypervisor specific format)
  - AutoYaST profile
- Public cloud
  - Image to run (public cloud specific format)

Infrastructure Artifacts (2/2)

- PaaS (example: Cloud Foundry)
  - Configuration file
- Container (example: Docker)
  - Configuration file (“Dockerfile”)

Deployment Options

- Manual install
- Automatic install
- Install from image
- Install from image and using CMS to configure
- Install using file

Deployment Overview

Deployment	GUI Editing	Build Tool
Physical image	Studio	kiwi
Appliance	Studio	kiwi
Automatic using AutoYaST	YaST Autoinstallation	autoyast
“Files” for CMS	None	CMS at runtime
Docker image	None	docker
Cloud Foundry Application	None	Cloud Foundry BOSH

Imaging

Imaging – The Cloud Deployment Way

- Approaches to Imaging:
  - Cloning an existing image
  - Take existing image like JeOS
  - Build new image
    - Kiwi (command-line, allows automatic building)
    - SUSE Studio (GUI)
    - Open Build Service

When and how to Image Build?

- When?
  - Manual, when ever the need arises
  - Automatic, when content has changed, for example packages get updated
  - DevOps workflow might contain triggered builds of images
  - Other users need a one time build that they reproduce and change
- How?
  - Command line for automatic builds
  - GUI for one time builds and non-experts

What to put into an Image?

- Complete, ready-to-run images (“Appliances”):
  - Contain complete workload
  - Allow running workload with minimal setup
  - Use for:
    - Same workload is deployed often
    - Bootup time is critical
- Base images:
  - Contain base system, perhaps runtime but not workload
  - Need customization at boot up for workload
  - Customization can be done using cloud-init, CMS
  - Use for:
    - Different workloads that needs customization
    - Large variety of different workloads
    - Short life workloads with frequent config changes

Image Customization: cloud-init

- At boot time user passes “user-data” to cloud launch tool
- Works with OpenStack, AWS EC2, Microsoft Azure etc
- Instance at boot time ask the “meta data server” for the “user-data”
- Usage:
  - Set up CMS
  - Set up hostname
  - Add users
  - Add ssh keys
  - Execute scripts

```
# Add groups to the system
groups:
  - cloud-users

# Add users to the system. Users are added after groups are added.
users:
  - default
  - name: foobar
    gecos: Foo B. Bar
    primary-group: foobar
    groups: users
    ssh-import-id: foobar
    lock-passwd: false
    passwd: $6$j212wezy$7H/1LT4f9/N3wpgNunhsIqtMj62OKiS3nyNwuizouQc
```

Image Customization: CMS

- CFEngine, Puppet, (soon SaltStack) part of SLES 12 Advanced Systems Management Module
- Install agent on image (Ansible, SaltStack (option): ssh+python) and configure system at run-time
- Can be personalized with cloud-init

Cloning

- Done by many customers today
- Needs to de-personalize:
  - Remove unique IDs
  - Remove passwords, keys

Take Existing Image

- JeOS for SLES 12
  - Both image and kiwi files available for download
- Dockerimages for SLES 12 and SLES 11 SP4
  - Part of SLES 12 Container Module

SUSE Linux Enterprise 12 JeOS

What is JeOS?

A subset of SUSE Linux Enterprise Server, designed for:

- Efficient cloud deployments
- Minimized physical deployments

SUSE Linux Enterprise Server with respect to:

- Certifications
- Availability of packages
- Subscriptions and pricing
- Policies for:
  - Maintenance
  - Support

Use Cases and Deliverables

Private Cloud Image

Deliverable: Ready-to-run (virtual) images

- For the major hypervisors on x86-64: Xen, KVM, Hyper-V, ESX
- Quarterly delivery based on SUSE Linux Enterprise 12 / SUSE Linux Enterprise 12 SP1 plus maintenance updates

Silver Image

Deliverable: Installable KIWI file

- Result is similar to the ready-to-run virtual image
- Customers build their Golden Images using KIWI

SUSE Linux Enterprise Server 12 JeOS

JeOS and Containers

	SUSE Linux Enterprise Server	JeOS (Guest VM)	Application Container "Docker"
Size	XXXL	M	XS
Download size (MiB)	2800	300	23
Typical on disk size (MiB)	1500	300_{qcow2}	102
Additional packages included	●		
Installer	●		
Docker host	●		
Cloud Init	●	❍	
Kernel, Bootloader, Init System, Registration	●	●	
Localization / Languages	●	●	
System update via zypper	●	●	❍
System Libraries	●	●	●
Download	download.suse.com	download.suse.com	SCC, SMT, SUSE Manager
SUSE delivery	SLES	JeOS	Container Module

Image Building

Build custom image with Kiwi

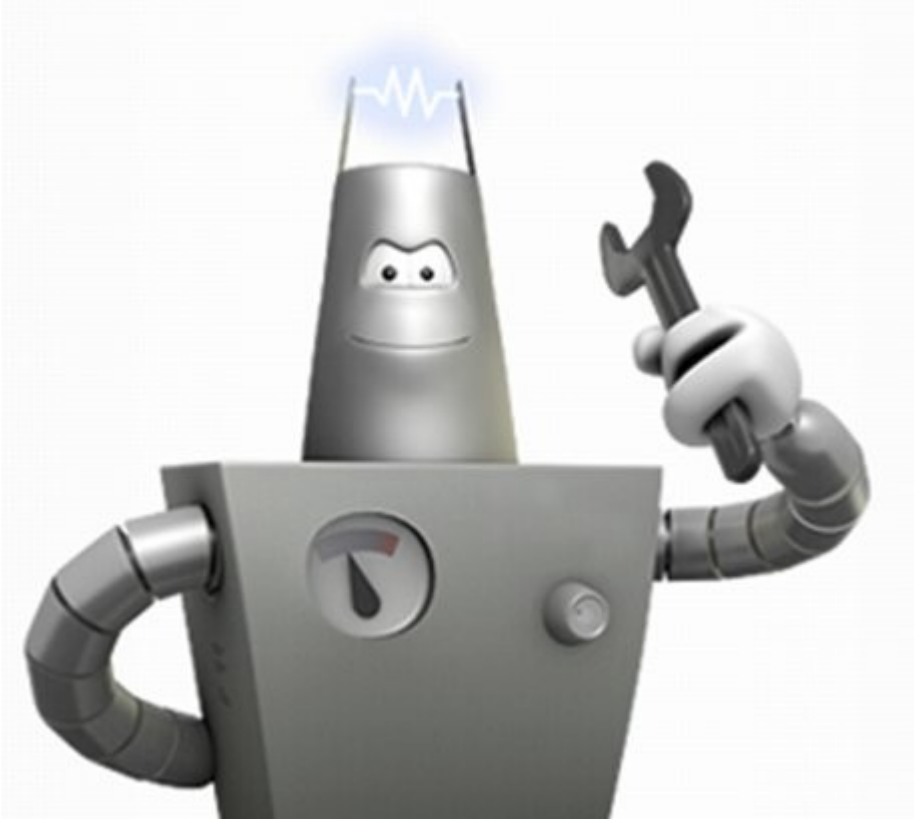
- Kiwi is supported in SLES 12
- XML configuration with many options
- User can start with JeOS kiwi file or export from Studio

```
<?xml version="1.0" encoding="utf-8"?>

<image schemaversion="6.2" name="LimeJeOS-SLE12">
    <description type="system">
        <author>Marcus Schäfer</author>
        <contact>ms@novell.com</contact>
        <specification>SUSE Linux Enterprise 12 JeOS</specification>
    </description>
    <profiles>
        <profile name="xenFlavour" description="VMX with Xen kernel"/>
        <profile name="xenFlavourHVM" description="VMX with default kernel plus xen modules"/>
        <profile name="vmxFlavour" description="VMX with default kernel" import="true"/>
        <profile name="docker" description="docker image"/>
        <profile name="vagrant" description="vagrant base box for libvirt"/>
        <profile name="netboot" description="compressed rootfs image for pxe"/>
    </profiles>
    <preferences>
        <type image="iso" primary="true" boot="isoboot/suse-SLES12" flags="overlay" hybrid="true
        <version>1.13.1</version>
        <packagemanager>zypper</packagemanager>
        <bootsplash-theme>SLE</bootsplash-theme>
        <bootloader-theme>SLE</bootloader-theme>
        <locale>en_US</locale>
```

Build custom image with SUSE Studio

- Easy to get started
- WebUI allows building and editing from any OS



SUSE Studio

Build once - deploy everywhere

- Build, test, deploy, and maintain software applications based on SUSE Linux Enterprise.
- Deploy them on physical hardware, in virtualized environments, or to cloud environments.
- Available as:
  - Online version at http://www.susestudio.com
  - Product for installation

Machinery: Migration to Cloud

- Inspection of system and …
  - Generation of kiwi file for building cloud images
  - Identifying workloads and creating Dockerimages for them
- Part of SLES 12 “Advanced Systems Management Module”

Related Technology

- SUSE Linux Enterprise Server 12 Modules:
  - Public Cloud Module:
    - Contains tools for manipulation of images and upload to public clouds
  - Advanced Systems Management Module:
    - Contains CFEngine, Puppet, Machinery
  - Containers
    - Contains Docker and tools

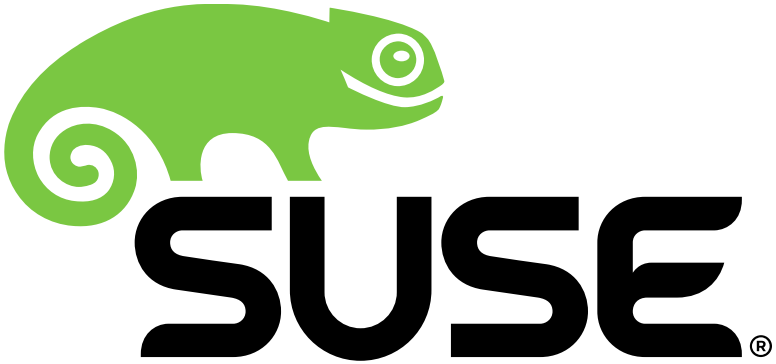
Some Related Presentations

- FUT19146 - Advanced Systems Management with Machinery
  - Wednesday, 9:00 AM - 10:00 AM – 11-Getuigenkamer
  - Thursday, 10:15 AM - 11:15 AM – 11-Getuigenkamer
- CAS20057 - Using Kiwi to Create Customized PoS Images
  - Wednesday, 5:00 PM - 6:00 PM – 13-Groot Secretariaat
- FUT20730 - Software Defined Everything - Management, Cloud, Containers and Storage
  - Wednesday, 3:30 PM - 4:30 PM – 6-Derkinderenkamer
  - Thursday, 10:15 AM - 11:15 AM – 6-Derkinderenkamer
- TUT20514 - SaltStack and SUSE: Systems and Configuration Management that Scales and is Easy to Extend
  - Wednesday, Nov 4, 9:00 AM - 10:00 AM – 6-Derkinderenkamer
  - Friday, Nov 6, 9:00 AM - 10:00 AM – 7-Rode Kamer

Multiple Endings...

Many different ways to build.
SUSE provides powerful solutions.
How are you going to build?

Thank you.



Corporate Headquarters
Maxfeldstrasse 5
90409 Nuremberg
Germany

+49 911 740 53 0 (Worldwide)
www.suse.com

Join us on:
www.opensuse.org