

# **Solaris Installation Strategy**

*Dave Miner*

dave.miner@sun.com

*Version 3*

*23. Mar. 2006*

# Table of Contents

1 Introduction.....	3
2 Problem Survey.....	4
2.1 SMB and Developers.....	4
2.1.1 Interactive Installation.....	4
2.1.2 Preinstalled Systems.....	7
2.1.3 Software Maintenance.....	7
2.1.4 Upgrade.....	8
2.1.5 Suites.....	9
2.1.6 Appliances.....	9
2.1.7 Performance.....	9
2.2 Enterprises.....	10
2.2.1 Interactive Installation.....	10
2.2.2 Preinstalled Systems.....	10
2.2.3 Software Maintenance.....	10
2.2.4 Network Deployment.....	10
2.2.5 Diskless Systems.....	12
2.2.6 System Replication, Availability and Disaster Recovery.....	13
2.2.7 Virtualization.....	13
2.2.7.1 Zones.....	14
2.2.7.2 BrandZ.....	15
2.2.7.3 Xen.....	15
2.2.8 Security.....	15
2.2.9 Third-party Vendors.....	16
3 Requirements Summary.....	18
4 The Way Forward – Caiman.....	20
4.1 Caiman User Experience.....	21
4.1.1 Initial Installation.....	22
4.1.2 System Configuration.....	23
4.1.3 Upgrade Installation.....	23
4.1.4 Automated Installation.....	24
4.1.5 Replication and Recovery.....	24
4.1.6 Diskless Systems.....	24
4.1.7 Software Management.....	25
4.1.8 Performance.....	25
4.2 Community.....	26
4.3 Caiman Project Roadmap.....	26
4.3.1 Live CD.....	26
4.3.2 System Configurator.....	26
4.3.3 Disk Partitioner.....	26
4.3.4 Interactive Installer.....	27
4.3.5 Automation and Replication.....	27
4.3.6 Software Manager.....	27

# 1 Introduction

Installation of Solaris has been a somewhat neglected technology sector in recent years, lacking in continuity of management and strategic focus. We have a large collection of installation technology and options for installation, but all of that technology can be difficult to use, and we often provide little direction to customers in how best to make use of its capabilities. Many of the installation technologies also have significant architectural issues.

The Solaris Nevada Marketing Requirements and Product Requirements documents<sup>1</sup> provide extensive background data on the need to improve ease of use in Solaris installation. Such improvements are seen as a gating factor to broadening the addressable markets for Solaris.

Our emphasis going forward must be on defining the ways in which our existing and future customers want to install and use Solaris, and aggressively implementing the key strategic technologies and practices that will support that usage while minimizing Sun's engineering and support costs. This paper attempts to provide that definition.

At this point, it is important to define the scope of the discussion which will follow. In order to make the problems in this space somewhat more tractable, we have chosen to separate the development of a Solaris strategy for Installation from the development of a strategy for Packaging and Patching. The two areas obviously overlap, together comprising the customer's software lifecycle, and the packaging technology provides a technical basis for implementation of installation, but we believe that a well-formed installation architecture can (and should) be independent of the packaging technology which is used in constructing the software objects.

Thus, this document neither assumes that Solaris will continue to use its existing packaging technology, colloquially known as "SVR4 packaging", nor that we will replace that technology with some other alternative. Discussion and evaluation of other packaging technologies has been ongoing both at Sun and in the OpenSolaris community for some time, and development of a Packaging and Patching strategy is underway. While this paper may reference some specific issues with those technologies in passing, the requirements and strategy proposed will not address those issues directly. If we do our job well, customers will be able to manage their Solaris software completely without needing to understand the details of the packaging technology we choose to use. In other words, packaging is mostly an implementation detail as far as installation is concerned.

This paper is structured in three chapters, respectively covering:

- problem area survey
- summarized list of requirements
- recommended strategy to address the requirements

---

<sup>1</sup> <http://solaris.eng/RELEASES/S11/Plans.html>

## 2 Problem Survey

Solaris is currently used primarily by enterprises to operate mission-critical data centers. These organizations have some problems with our existing technologies, which seem to them to be increasingly dated and lacking consistent, cohesive integration. In addition, new technologies such as ZFS, Zones, and Xen will be used by these organizations in ways that will require changes to the installation processes we offer. These technologies do not merely represent a problem, though; they, in concert with other recent developments such as SMF, also present significant opportunities for us to re-examine the assumptions on which Solaris installation has been constructed and rebuild it on a more fundamentally sound footing.

Enterprises account for the majority of Solaris revenue, and thus meeting their evolving requirements is crucial to maintaining our base. At the same time, the Small/Medium Business market is nearly as large as the Enterprise market, and growing more quickly, yet Solaris is currently not a significant player in this market. We believe these customers have substantially different needs than the enterprise customers. Many of them may only be closely familiar with Windows or some of the Linux distributions, and will expect a similar level of ease of use and polish in any operating system that they consider adopting. Their ability to quickly install a basic server and corporate workstations, and upgrade both, will be a crucial part of their willingness to consider adopting Sun's technology.

Looking further, the foundation for a customer buying Solaris is that the software they need to operate their business runs on it. For this to be true, Sun must make it appealing for developers to develop on, and for, Solaris. Developers today are familiar with Windows, and most will have worked extensively with other Unix-like offerings, such as MacOS and the major Linux distributions. The expectations of these users for ease-of-use will therefore be high, and their patience short. For them to consider adding Solaris to the set of operating systems they regularly interact with, they must be willing and able to install the product, and for them to do this we must have an extremely simple and appealing installation experience that works well on a laptop that has other operating systems on it. While these users are not identical to the SMB, their background is similar enough that we will consider them as mostly the same for the purposes of the discussion which follows.

The remainder of this section considers Solaris installation functionality from the point of view of two different customer profiles: the Developer/SMB, and the Enterprise. Along the way, specific architectural issues with the current features are also discussed.

### 2.1 SMB and Developers

Since few of the potential customers in this space are familiar with Solaris, and the scale of their operations is not necessarily large enough to invest significant time in learning complex automation techniques, our analysis is focused primarily on the interactive installation experience and the tools provided for simplifying other aspects of software maintenance.

#### 2.1.1 Interactive Installation

Customers in the SMB and Developer segments expect an easy to use and appealing

graphical installation experience. As it's usually their first contact with the software, the installation process is really the face of the software, and will strongly affect the user's perception of the entire product. First impressions do matter, and negative impressions can be hard to overcome, especially if they are so negative as to inspire the user to walk away from Solaris..

In order to gauge our position relative to the competition in the SMB/Developer market, Solaris marketing has performed a comparative analysis of Solaris interactive installation versus the competition<sup>2</sup>. Generally, the results can be summarized thusly: Solaris installation is ugly, slow, and difficult. Anecdotal data collected since the Solaris 10 release supports this analysis. Such a perception undoubtedly hinders Solaris' ability to penetrate some key markets; while we have great technology once you have a running system, getting to that point is more painful than anyone who is not a committed Solaris user may be willing to bear.

The most obvious issue here is a dated visual design. An updated, consistent visual design that uses Sun's branding and promotes our values and image is highly desired.

In terms of ease-of-use, the Solaris installation experience leaves much to be desired. As noted, our own studies show that the experience is not as easy as with our competition. It's interesting to observe that the core installation experience dates essentially to Solaris 2.1, and has not been especially updated since that time. There are many notable problems:

- We ask the user for a great deal of configuration information up front, before we've actually determined whether it's needed to complete the installation tasks. This creates user frustration, as it's a distraction from the primary goal and different than the model they encounter with any other operating system.
- We use terminology throughout the install process which is not readily comprehended by users who are not already experts in Solaris.
- We use outdated networking technology (RARP and Bootparams) by default, rather than contemporary network protocols, and thus are often unable to automatically determine configuration attributes that are easily discovered by our competition.
- Users do not clearly understand when in the installation process we begin actually modifying the system, and we do not clearly identify the point at which their attention is no longer required.
- We are unable to cope well with multiple-OS installation environments, which are common among the developer base. The introduction of GRUB-based booting in Nevada and Solaris 10 1/06 has addressed one of the issues, but we are still handicapped by our inability to non-destructively resize other OS partitions on x86, the usual answer being to boot Windows and use Partition Magic to do the resizing, or use open-source tools such as `qtparted` on Linux. Which leaves one to wonder how many potential users just decide at this point that Linux is more what they want, anyway.
- We don't include the right set of initial configuration tasks, such as an initial user account, that are commonly provided by competitors. This results in an installed system which boots, and can be logged into as root, but it's then up to the user to hunt around and find a tool (or, more likely, edit the configuration files directly due to our paucity of

---

<sup>2</sup> <http://webhome.sfbay/solarismarketing/solarisinstall/install.html>

- tools and poor integration of those that exist into the desktop) to create a usable account.
- We're deficient in the user experience for selecting the software to be installed. The "metaclusters" that we present as choices are relatively meaningless to these users and provide little value, as there is not a particularly significant difference in size of installation between the Developer, and Entire Release metaclusters. We view the End User metacluster as essentially meaningless at this point – Solaris desktop users who aren't developers at some level are few and far between these days. However, even if we improve the metacluster definitions in some way, we'll still find that advanced users want to customize the software further. In that context, we're even worse off, as the installer is unable to automatically resolve dependencies between packages, such that the user is forced into a tedious game of "fetch a rock" to work out dependencies if any level of customization of the installed software is desired. Better package dependency declaration and management are desperately needed.
  - On x86/x64 hardware, we continue to have issues with hardware support. This isn't directly an installation issue, but strongly affects the customer's initial impression - wading through an entire install (especially with the deficiencies already noted), only to find that some of the system hardware is not supported, can only be a dissatisfying experience. Several separate issues exist in this area:
    - 1) We should be able to tell the user up front whether the system hardware will be completely supported. We have recently made available the Installation Check Tool<sup>3</sup> to help in addressing this, but it is a separate CD image which must be downloaded, burned, and booted to provide the answer.
    - 2) We exacerbate the hardware support problem by only installing the drivers required by the system when it's being installed, unless the "Entire plus OEM" metacluster is installed, in which case we do install all drivers. This means that if the user later adds a different type of network interface or storage device, the user may be faced with a tedious process of locating the installation media, finding the correct package, and installing it. This model dates to the days when disk space was expensive, but on current builds of Nevada the difference in space required is approximately 50 MB, which is less than a 1% difference.
    - 3) Customers often need to provide additional drivers to be used during install, most commonly these are either network or storage drivers. The interactive install does provide a guided utility to add drivers to the install session, but for Jumpstart installs, the customer must instead undertake a complex process of generating a modified miniroot archive with little documentation.
  - One significant complaint with interactive installation from CD/DVD is that we default to booting from the media; this presents a problem if the media remains inserted in the drive after rebooting, as the system ends up rebooting back into the installer. As a result customers who have walked away may believe that the system did not install successfully; at the very least they have to initiate yet another reboot to start the software installed. Other vendors generally default their installation media to booting off of the hard drive instead of the CD to avoid this problem. Solaris should do so as well.

---

<sup>3</sup> [http://www.sun.com/bigadmin/hcl/hcts/install\\_check.html](http://www.sun.com/bigadmin/hcl/hcts/install_check.html)

## 2.1.2 Preinstalled Systems

When an SMB or Developer buys a system from Sun, they expect to have the same "plug and play" experience they get when buying a system from Dell or other vendors who are leaders in this space. This forms their immediate impression of the quality of the product they just bought.

Sun has long shipped our systems with a pre-installed copy of Solaris imaged onto the disk, which allows the customer to unpack the system, power it on, and at most answer the usual configuration questions, so we are somewhat on the right track. However, we have received a great deal of feedback from NSG that the current experience with preinstalled systems is not adequate. Essentially, the problem here is merely a subset of the interactive installation problem - the software is installed, but the configuration process is still archaic.

Some of the issues here are architectural in nature, as we require a reboot to configure some system parameters. Solaris currently requires a reboot if the timezone or locale is changed in order to ensure that all processes are operating with the current values. NSG also performs custom hardware configuration procedures on some systems, which may require a reboot and are not integrated with the Solaris configuration procedures, resulting in potentially two reboots prior to the system being ready for use.

## 2.1.3 Software Maintenance

One of the significant deficiencies in Solaris compared to our Linux competitors is our ability to easily install additional software after the initial installation. The SVR4 package commands have been enhanced over the years to provide support for downloading packages over the network and verifying package signatures. However, we provide no tools to create, manage, and use software repositories, automatically update software (other than CNS's Software Update, which so far only provides patches to already-installed packages), and resolve dependencies. Contrast this to the capabilities of Debian-based systems, the RedHat and Novell update services, or even Windows Update, and it's clear that we are well behind the leaders here.

One important consideration is that the need here is not just to supply patches - it's also to supply completely new sets of packages for a particular subsystem, such as Gnome. This requirement is reflected strongly in the Solaris/OpenSolaris development community. Tools such as BFU, Blastwave's pkg-get, and scripts provided by the various Solaris consolidations such as X server or JDS to update their components, are all evidence of unmet requirements in the core packaging and software maintenance system.

This lack of strong support for additional software installation also increases the stress around the initial installation - users quickly learn that if they don't install the correct set of software for their system's usage, they will be faced with a painful process in locating the additional packages they need from some copy of the installation media. Thus, they really feel betrayed if they don't get it right from the very beginning. This also contributes to the perception that our install is slow, because it means that often more software than strictly necessary is installed before the system is made available to the user. Newer Linux distributions such as Ubuntu provide a superior experience in this respect, installing one CD's worth of software to achieve a running system, and relying on fairly easy-to-use tools

such as Synaptic to acquire additional software as needed.

### 2.1.4 Upgrade

At present, Solaris supports two different operating system upgrade technologies:

- "Standard" upgrade, wherein the system is booted off the release media and the operating system image is upgraded in place
- "Live" upgrade, in which the system remains running on its installed operating system, while the upgrade is carried out on a separate copy of the operating system.

Live Upgrade has compelling advantages for the customer, as it provides the ability to revert back to a prior version should problems be encountered with the upgraded system, and also minimizes the downtime required for an upgrade, limited merely to the reboot required to transfer over to the upgraded system. However, comparatively few customers use Live Upgrade, the two main reasons appearing to be lack of awareness of the feature, and lack of planning to use it, since it currently requires additional disk space to be allocated for the copies of the system. Since Live Upgrade is a feature not seen on Windows or Linux platforms, customers in this category are highly unlikely to recognize the need to plan for it. We need to compensate by automatically planning ahead for the customer.

However, one significant barrier in the SMB market is that Live Upgrade's user interface is not simple: it's completely command-line based, and the commands involved in setting it up can be complex to comprehend. A simplified, FMLI-based interface was developed and is still shipped, but is effectively crippled because it has not been evolved to support newer features, so at this point it is merely an attractive nuisance which can only cast a negative light on the Live Upgrade technology.

An additional issue with Live Upgrade is that the boot environments which are not running can be managed in only a limited fashion; package manipulation is possible thanks to flags which specify the root of the filesystem on which to conduct the operations, but packaging scripts are greatly complicated by the need to account for whether the packages are being installed into a running system or an alternate root. Most other administration operations are not possible on alternate boot environments, and to enable them would at present require specific changes in any utilities which the administrator uses for those operations, but this is not a scalable solution. This issue also rears its head anytime a customer attempts to write finish scripts for Jumpstart installations.

A final issue with Live Upgrade is somewhat inherent in its architecture: the upgrade is necessarily running on an older version of Solaris, which means that the upgrade code must be implemented to the API's offered in Solaris as of two or three releases prior to the release being installed, depending on our currently supported upgrade story. This limits our ability to quickly take advantage of new features in the installer without requiring significant extra design and implementation work to deal with the absence of features in the older releases.

### 2.1.5 Suites

Sun's software strategy at present emphasizes selling customers integrated solution suites, rather than a selection of components which must be integrated together; making headway in the Developer/SMB market with both Solaris and our layered software products would seem to require such an integrated offering. This strategy, however, is hampered by a lack of an integrated install experience between Solaris and the other software products. The current Solaris installer has some ability to install additional products along with the Solaris installation, but this capability is not widely used and is not regarded as meeting current requirements for layered software installation.

There is currently a project known as EZOffering<sup>4</sup> which attempts to provide a simple, integrated install for some of the JES suites, consisting of a bootable DVD with Solaris, a specific JES suite, and any required Solaris patches, bundled into a simplified install requiring a minimum of user input to achieve a usable installation of the suite.

Longer-term, the Purple Haze<sup>5</sup> project is building a new installation engine initially targeted at the needs of the JES products, though with the ambition of providing a unified platform for installation of all Sun software.

There is also presently an effort underway, led by the xDesign group, to design a unified look and feel for software installation at Sun. This is separate from, yet closely overlaps, the Purple Haze project. We are participating in this effort.

### 2.1.6 Appliances

One potential avenue for Solaris to make headway in the SMB market is as the core engine for specialized appliances for storage or networking functions. A contingent with the OpenSolaris community has established the Appliances community<sup>6</sup> to support developments in this area. We do not have detailed installation requirements for this space, though discussions with Sun projects investigating such products indicate their desire to use features such as Flash archives and Live Upgrade as part of their software lifecycle.

### 2.1.7 Performance

Performance is always an important attribute of a software system, and it is certain that performance of installation is a factor in users' initial perception of the system performance. At present, we do not have systematic data comparing installation of Solaris with competitive operating systems, though the perception is that we are slower. Anecdotally, we observe relatively poor utilization of the installation media when installing from CD or DVD, which can noticeably affect performance as the media is repeatedly spun up and down to transfer packages. We also observe that the existence, and continued popularity, of the ON consolidation's BFU utility provides evidence of unmet performance requirements in installation among some segments of the developer community, though other factors contribute to BFU's continued usage.

There has long been concern around the SVR4 packaging technology performance; during

---

4 <http://twiki.sfbay.sun.com/bin/view/MDE/SDPProgram>

5 <http://install-arch.sfbay.sun.com/Wiki.jsp?page=PurpleHaze>

6 <http://www.opensolaris.org/os/community/appliances/>

Solaris 10 the install team attempted to replace the `/var/sadm/install/contents` text file with a database, but the results were so poor that the project was backed out. Other recent proposals have been put forth to address this problem, but we have not undertaken a systematic analysis and investigation of alternatives. A performance analysis at the packaging level should be undertaken to ascertain whether there are any low-hanging fruit to be gathered; this seems rather likely, as the complexity of the packaging tools has grown greatly, especially with the introduction of Zones.

## **2.2 Enterprises**

While Enterprise customers would undoubtedly appreciate ease-of-use improvements to interactive installation, the improvements they desire are reflective of their more extensive knowledge of Solaris. Enterprise customers quickly learn that the more efficient way to install Solaris is over the network, from centralized installation servers, using automated installation technology. They are also heavy users of virtualization and other technologies which can increase their operational efficiencies and minimize their IT expenditures.

### **2.2.1 Interactive Installation**

For Enterprise customers, a more attractive installer is desirable, but their principal issue with the interactive experience is the poor support in selecting the software to be installed, a problem already covered in the SMB discussion of interactive installation on page 4, and which we'll expand upon later in Security on page 15. One feature which has been requested is the ability to generate a Jumpstart profile directly from an interactive installation session, either in addition to or instead of performing the actual installation.

### **2.2.2 Preinstalled Systems**

From an enterprise customer perspective, the preinstalled Solaris that is shipped with systems is only rarely useful. Most of these customers define a customized standard load of software for their systems, which is rarely the entire current release of Solaris; these tend to lag significantly behind the version of Solaris which we preinstall at the factory.

One idea Sun might explore in this realm would be a way for customers to provide a standard load that we could preinstall for them, but otherwise there appears to be very limited demand for preinstallation from this customer base.

### **2.2.3 Software Maintenance**

Enterprise requirements for software maintenance are largely similar to those of in the SMB market, with the additional requirement that these customers will often wish to set up and manage their own repositories of software and patches, filtering and customizing the contents to their requirements rather than absorbing it directly from vendors such as Sun.

## 2.2.4 Network Deployment

Solaris has long supported network installation, with a network boot architecture founded on OBP's support of network protocols such as RARP and tftp to acquire the boot loader, followed by use of Bootparams and NFS to locate, boot, and configure the operating system.

During the early Solaris 2.x releases, an automation framework known as Jumpstart was developed to provide for hands-free installation. Beginning with Solaris 8, this support was augmented with the use of DHCP, which provided the equivalent of the RARP and Bootparams functionality for acquiring the network and operating system configuration parameters, without the inherent restrictions of the earlier protocols; we also added support for the x86 platform's PXE function, which provides a subset of the functionality of OBP on SPARC. Further, in Solaris 10 (and Solaris 9 updates), the WAN installation project was implemented for SPARC platforms, providing a means to install the operating system over public networks, replacing the usage of tftp and NFS with HTTP-based downloads of the boot loader and operating system image, but limited to installation based on flash archives. Most recently, Solaris Nevada (and Solaris 10 Update 1) replaces the proprietary x86 boot loader with GRUB, which, among other features, simplifies the x86 network boot experience.

In order to manage Solaris deployments, we provide customers with bundled scripts such as `setup_install_server` and `add_install_client` to perform the rudimentary tasks of making an installable operating system image available on a server and configuring it to support a particular client (or set of clients). When DHCP is used, it's also necessary to separately manage data in the DHCP service to provide the full network installation function. Instead of using the bundled tools, many customers use the JET toolkit developed by Sun Professional Services, which only recently became supported as part of the N1 provisioning product line; JET is attractive because it provides higher levels of functionality built upon the Jumpstart foundation. Some of the N1 products provide other options for provisioning Solaris systems.

In this space, there are several problems. The most obvious is that we have several technologies for network provisioning, but little coordination of product direction. The mere existence of JET as a separate, unsupported toolkit was a bug, which fortunately N1 seems to be addressing by taking ownership of it; however, there is as yet little or no communication between the N1 team and the Solaris install team on the architecture and possible requirements for enhancement to Jumpstart that JET's considerable experience and large body of functionality would suggest are likely to exist. Similarly, there has been little recent communication between the other N1 provisioning products and the Solaris installation team, and that cannot bode well for the customer experience. Clearly, communication channels need to be established between these groups, with a goal of providing an integrated user experience between Solaris and the N1 products.

Focusing more closely on the bundled Solaris installation pieces, there are a few issues here which need to be addressed:

- The bundled tools do not provide full task integration, which would make it easy to set up a full network provisioning environment on a single server. Additionally, they do not

provide or consume stable interfaces to alternate DHCP services and other potentially replaceable components.

- We have received feedback from several large customers and partners that they would like to see a re-factoring of the deployment technology to support use of the WAN-style configuration capability, but without the requirement to use flash archives and to instead use standard network installation images. The reason behind this is that, in many customer networks, the responsibility for managing DHCP service lies with a different group than that charged with managing the Solaris infrastructure. This introduces considerable difficulty in supporting network installations, as even if the group supporting DHCP can be convinced to provide the necessary parameters for Solaris installation, the frequency of required updates and coordination involved for specific systems tends to increase the cost of supporting Solaris.
- Support for WAN-style installation on x86/x64 systems is a glaring omission. There is no intrinsic reason why it's only interesting for SPARC customers, and would seem to be a significant strategic technology for x64 blades, a potentially high-growth business. This support was not originally provided because the primary customers were only interested in SPARC support, and the boot subsystem on Solaris x86 at the time was too complex to contemplate the additional effort. While the OBP on SPARC simplifies some of the implementation of such support, a fair facsimile seems possible with appropriate extensions to GRUB, which has since become our boot subsystem on x86.
- The architecture around the `setup_install_server` and `add_install_client` tools needs to be reconsidered. At present, these utilities are provided bundled on the media for a particular release, and are not actually installed on the system. This would seem to make sense, in that the tools are only needed and useful when installation media are available. However, the maintainability of this choice is poor. By requiring use of tools which are effectively frozen in time when the release is constructed, it is virtually impossible to effectively fix problems found with them, and it is also impossible to evolve them to support evolutions in later versions of Solaris. As a result of this architecture, projects such as the Service Management Facility (SMF) have been forced to implement sub-standard user experiences in order to avoid breaking `netinstall` tools from ancient releases. These tools need to be re-architected so that they may be released as part of the system and evolve in unison with the rest of the system.

## 2.2.5 Diskless Systems

Solaris has supported diskless clients, which access their filesystems exclusively over the network using NFS, for about twenty years, though in recent years the need for continuing support has been questioned, with some justification. The diskless client support in Solaris was designed to provide low-cost desktop clients, with centralized administration and no local storage, but the Sun Ray appliance has largely usurped the low end of that market with a generally superior solution. Additionally, the costs of desktop workstations have plummeted to the point that there is little need for a solution at a higher value point than the Sun Ray. The only significant exception to this point has been the need in some secured environments to retain centralized installations of the operating system, but this historical requirement has not been recently explored for its continued relevance.

More recently, the rise of horizontally-scaled datacenter architectures and grid computing have created demand for a new type of "diskless" system. In these architectures, the systems are intended to be fairly low-cost, replaceable units, easily re-purposed to various applications as the demand dictates. However, the systems have sufficient memory and network bandwidth to support a model where the operating system image is downloaded to RAM and executed directly from there, making them more resilient against failure of the server and network which supply the operating system. We do not support this newer model at present, but it seems the time is ripe to investigate whether the traditional diskless-over-NFS architecture should be replaced by this new architecture, or whether the two architectures are both needed going forward.

The introduction of iSCSI provides another alternative to our traditional model, the impact of which we have not yet assessed.

Diskless client maintenance suffers from similar issues as Live Upgrade boot environments as discussed on page 8. Whatever diskless model(s) we choose to support going forward, those maintenance issues must be addressed.

### **2.2.6 System Replication, Availability and Disaster Recovery**

One of the significant requirements enterprise customers express is the ability to quickly and easily replicate an existing system image, often to provision additional capacity for horizontally-scaled applications or to recover a system when various sorts of disasters occur. Solaris provides the "flash" installation method in response. Flash is essentially a `cpio` archive of the system, which can be used as an installation medium both by Live Upgrade and the standard installer. One obvious problem here is that flash archives are an installation medium, not a directly usable image - the result is that customers also need the system to either already be running (if using Live Upgrade) or need access to an installation image (CD/DVD or network) in order to actually use the images.

In most cases, what is preferred appears to be something more along the lines of AIX's `mksysb` utility, which creates a bootable image that can be used on its own to install the system and application software. Currently, Sun provides blueprints articles which explain how a customer can construct custom installation media based on the standard Solaris CD/DVD to implement this capability. Unfortunately, these blueprints can easily become outdated as we evolve the system architecture, a situation already seen on x86 platforms with the introduction of GRUB booting and presumably to be repeated on SPARC as the boot architecture evolves. The obvious need here is to design and implement supported tools so that all customers can easily create their own custom restoration/replication media without resorting to unsupported documentation and tools or homegrown solutions, which have been created by many large Solaris customers.

It's also been suggested that we might provide automatic system replication services, wherein a system might have a designated "spare" system to which it automatically mirrors its software. Requirements for such a feature have not been investigated at this time.

Sun Cluster's requirements for Solaris installation also need to be considered in this space, but have not yet been investigated.

## 2.2.7 Virtualization

The increasing variety of virtualization technologies presents challenges for Solaris installation. Each virtualization technology exposes a different model of the system: a complete virtual machine in the case of VMware and Xen, lighter-weight containers implemented directly by the operating system in the case of Zones and BrandZ.

### 2.2.7.1 Zones

Zones, as we have all experienced, placed a tremendous number of new requirements on the software management technology.

Prior to the introduction of Zones, a system image consisted of a single instance of Solaris, with at most some set of alternate boot environments due to Live Upgrade or some diskless clients; in each case the software management functionality was restricted due to simplifying limitations imposed. Now, we are faced with the need to install, patch, and upgrade multiple instances of Solaris in lockstep. With Solaris 10 1/06, we have provided Ashanti, an limited solution for upgrading a system with installed zones using patches, and the Zulu project will provide the permanent solution using packages in another Solaris 10 update.

However, even with the massive investment embodied by those projects, significant functionality related to Zones is still missing, most especially the integration of Zones support with Flash installation. At present, Flash is unaware of the existence of Zones on a system, and as a result in most cases cannot correctly replicate a flash archive containing Zones onto a newly-installed system. This limitation is a significant stumbling block to customer deployment of Solaris 10 and Zones in some large accounts.

Another problem raised by customers with regard to Zones is the lack of a fully-automatic provisioning and configuration capability. As noted in the Large scale deployment section, customers extensively use Jumpstart profiles to automatically install systems to a known configuration. Customers have requested a similarly automatic provisioning and configuration capability be supported for Zones.

The other significant known issue with Zones is the great deal of flexibility allowed to the administrator in determining which software is inherited from the global zone into the non-global zones rather than being installed directly into the non-global zone. This presents a major problem for the packaging tools in accurately installing and updating zones because the boundaries for inheritance of software from the global zone are specified based on directories, but that specification is orthogonal to the packaging boundaries. This seems to be a significant architectural error. One of the reasons diskless clients were successfully supported over the years without massive surgery to the packaging tools and the packages was that the packaging boundaries were established based on the architecture of diskless client support, which specified that each client had its own writeable / filesystem and a shared, read-only /usr filesystem. Several notable problems occur as a result:

- The packaging tools must create a replica of the entire packaging database inside the zone, even when the software itself is merely imported by read-only loopback mounts. This complicates and slows down zone creation, as well as wasting space.

- The packaging and patch tools must resort to horrible kludges to deal with class action scripts inside sparse zones. Basically, they're just not run at all, which means that there's essentially no real guarantee that the packaging database inside the zone is ever correct. In some notable cases, such as the Java VM, we know it's not.
- Users are hamstrung when we publish a package as "allzones." If you don't want Sun's SSH because you're planning to use someone else's, then that's just too bad. You have to accept our SSH in all of your zones because we know better than you, or remove it completely from the system.

We need to reconsider the sharing requirements of sparse zones and possibly rewrite the rules for Solaris packaging to address these issues cleanly.

A final, recent issue with Zones is raised by the introduction of zone migration. This feature allows a zone to be detached from one system and moved to another. As part of this process, a software inventory is taken, both before & after the move, to determine whether the destination system is capable of running the zone, and the user is informed of any packages or patches are required to be installed to synchronize the zone with the destination system. Going forward, we should extend this feature so that if the destination system contains newer packages or patches than the source system, the zone can be automatically upgraded as part of its attachment to the destination system to the same level as that system.

### **2.2.7.2 BrandZ**

BrandZ, as presently constituted, presents few new issues for software management, in that it merely presents zones which do not require any Solaris software management because they contain no Solaris software. There has been discussion of leveraging BrandZ to support Zones containing mixed versions of Solaris; should that discussion turn into a supported feature, there would be clear impact on the software management capabilities, as we would need to differentiate between versions of Solaris Zones and manage multiple versions of Solaris sensibly, though the latter problem is similar to managing Live Upgrade boot environments or diskless clients which are running a different Solaris release.

### **2.2.7.3 Xen**

The installation requirements around Xen (and VMware) are presently unknown. As a client operating system, Solaris would presumably be relatively unaffected, as the virtual machine technology should isolate the operating system from the fact that it is running inside a virtual machine. As a host operating system, however, some of the same issues as Zones may apply. For instance, will customers desire to upgrade or patch virtual machine environments in sync with the host operating system? We need to initiate a discussion with the Xen team on the software administration requirements in order to avoid a repeat of the Zones experience.

## **2.2.8 Security**

Enterprise customers have expressed several security concerns related to installation.

The advent of regulatory requirements such as Sarbanes-Oxley has led to increasingly stringent audit requirements from customers with regard to the security of their systems.

These requirements often translate into completely hands-free duplication of a specific configuration, with a minimal (ideally, zero) window of exposure to unauthorized access to the system.

Packages are increasingly offered by, and obtained from, a variety of sources, and there is a basic requirement to verify that the package came from a trusted source, so customers would like to have the ability to authenticate the package creator.

In order to ensure that network installations are not compromised, especially when installing software over public networks, it's often necessary for both clients and servers to authenticate each other, and possibly encrypt communications, including the actual software to be downloaded and installed.

Installation and upgrade present a potent security hole given the way we construct the installation media and the filesystems installed. We actually encourage this to some extent: the standard procedure for recovering from a lost root password is to boot the system from the installation medium (CD/DVD or network), mount the root filesystem, and clear the root password out of `/etc/shadow`. It would be desirable to provide some level of defense against an attacker who does gain physical access to a system, so that it's not a free license to access the installed system and compromise it.

Minimization is a common practice for addressing some classes of security issues, the theory being that software which isn't installed can't be used as an attack vector. Arguments both for and against this practice exist, we will not recap them here, suffice to say that the requirement exists and is mentioned by a high percentage of enterprise customers; discussions with IT personnel at these sites tend to raise this as an issue 50% of the time, or more. The Core and Reduced Networking clusters were intended to address this requirement to some extent, but feedback from customers indicates that they still regard them as too large and that there are items not strictly required, occasionally due to poor granularity of the packages, such as mixing client and server support for a protocol in the same package. These customers continually raise questions about what configurations we will actually support. This will probably be an ongoing problem without a permanent resolution, because each customer has a different idea of what "minimal" means, but nonetheless Solaris should provide a well-defined minimum that is backed by a support commitment. We must note, though, that an aggressive minimization strategy is often incompatible with our patch model; customers can easily end up requiring additional packages to be installed in order to apply a needed patch, due to patch boundaries being orthogonal to package boundaries. This issue must be addressed by the Patch strategy.

### **2.2.9 Third-party Vendors**

Enterprise customers will often select software from other vendors to augment the functionality of Solaris, common examples include Veritas, CA, or EMC. We have historically not made it easy to integrate that software into the Solaris installation experience, and indeed the customer is more likely than not to experience breakage in those components when Solaris is upgraded. Not all of those issues are attributable to Solaris installation software, but in some cases they are. Overall, the net effect of this situation is that it often results in delayed customer adoption of new Solaris releases and slower rollout of patches due to additional testing requirements, both of which have negative effects on

Sun's business. Providing formal interfaces by which these third parties can integrate with the Solaris software lifecycle would be a positive step towards providing the customer with a desirable Solaris experience.

A similar issue exists in relation to the cross-platform enterprise-management systems used by many customers. Enterprise customers tend to favor management systems which work across all of their platforms, and almost all enterprise customers have at least three operating systems installed. Providing stable interfaces for software inventory and management that can be used by those vendors is an area for further investigation.

### 3 Requirements Summary

As should be apparent at this point, Solaris installation is a large, complex system with a large set of needs across a very diverse customer base. Analysis of the issues discussed in the previous chapter leads to the following requirements:

- 1 The user interface must present a modern graphical look and feel, consistent with Sun's branding and consistent with other Sun software products.
- 2 The user interface must support both graphical and text environments.
- 3 The user interface must be simplified to present only those questions absolutely necessary to correctly install the software. Non-essential questions should be deferred until the system is installed, or, preferably, eliminated entirely.
- 4 The user interface must not require additional user input once the user has selected the software to be installed and the location to which it will be installed and initiated the actual installation process.
- 5 The user interface must use clear, non-technical terminology whenever possible.
- 6 The user interaction must be similar across both graphical and text environments.
- 7 The user interface must support both initial installation and upgrades from prior releases of Solaris
- 8 Initial installation and upgrade must be supported in both off-line and live forms
- 9 Initial installation and upgrade must be supported from manufactured media such as DVD's, or images of same.
- 10 Initial installation and upgrade must be supported over both local-area and wide-area networks, including through firewalls.
- 11 Initial installation and upgrade must support authentication of both clients and servers in network transactions, and encrypted communications between clients and servers.
- 12 Initial installation and upgrade must support integrity checking of the software to be installed and authentication of package creators.
- 13 Initial installation should offer to verify support of the system's hardware before attempting to install the software.
- 14 Installation must offer the option to integrate non-bundled drivers into the system for use during the installation process, and install those same drivers into the finished system as part of the installation process.
- 15 Installation must support coexistence between Solaris and other operating systems on the same system.
- 16 Installation must not require the use of non-Solaris tools to create space for Solaris on the system's disks when other popular operating systems are also installed.
- 17 Configuration interfaces used during the installation process should be similar to the interfaces used after installation for similar tasks.

- 18 Initial installation and upgrade must offer the ability to execute without the requirement of any interactive input during the process.
- 19 Installation tools must offer the customer the ability to create customized installation media.
- 20 At the completion of installation, the user should have an initial, non-root account available for login.
- 21 Equivalent installation capabilities must be offered across all supported hardware platforms.
- 22 Configuring services for network installation on a single server must be one integrated, self-contained user task.
- 23 Initial installation and upgrade of virtual systems such as Zones and Xen virtual machines must be supported, and must provide similar automatic installation capabilities as for physical systems.
- 24 Initial installation must allocate system resources such as disk space in such a way that customers will be able to later upgrade with either the off-line or live method.
- 25 Upgrade should provide the ability to revert the system to its pre-upgrade state. A failed upgrade must do so automatically unless otherwise instructed by the user.
- 26 Users must be able to easily install additional operating system software after initial operating system installation, with any software dependencies automatically resolved and installed.
- 27 A graphical interface for managing system software must be provided.
- 28 Users must be able to create and manage repositories of installable software packages, with those repositories discoverable and usable by the tools used to install software packages.
- 29 Users must be able to easily upgrade defined subsets of the system software (for example, the GNOME release), without requiring a complete system upgrade.
- 30 Upgrades must be sufficiently fast that the time required when using modern LAN technology will allow daily upgrades by developers without significant impact on their productivity.
- 31 Configuration parameters required to perform an installation should be collected automatically, using the network if possible. The user must be prompted only when necessary, or when specifically requested by the user.
- 32 Installation of Solaris must be capable of being integrated with installation of layered Sun products and software products of other vendors.
- 33 The installation and packaging technologies used by Solaris must be released into and developed under the OpenSolaris program.
- 34 Preinstalled systems must not require a reboot between initial power-on and user login.

## 4 The Way Forward – Caiman

In moving Solaris installation forward, we face the challenge of demonstrating immediate progress, as some of the deficiencies are viewed to be a pressing problem, but the preceding sections also demonstrate that there are a broad set of requirements which are not met by the existing architecture. This paper has largely ignored the short-term need to enhance Solaris installation to support current technologies such as Zones and ZFS – those requirements have been catalogued and prioritized by the Install team over the past several months<sup>7</sup>.

Our proposed approach to evolving the Solaris installation technology is to separate the two problem spaces, implementing business-critical enhancements to the existing installation technology as necessary, while in parallel developing a new install architecture designed around the requirements elucidated in this paper. The goal, obviously, is to rapidly develop the new installer and divert most enhancements to it as soon as possible. The remainder of this paper focuses on producing the new installer.

The new installer's proposed code name is Caiman<sup>8</sup>; at a high level, we have two goals for Caiman:

1. Create a competitive, inviting interactive installation experience
2. Upgrade Solaris' automated installation capabilities to be the best in the industry

Achieving the first goal removes a barrier to acceptance of Solaris in the SMB/Developer market, while achieving the second will help retain our historic strength in the Enterprise market.

The Caiman program is founded on a small set of basic principles which guide the elements of the strategy and the design of the projects which will be undertaken:

1. The user's goal at installation is to place Solaris on the system as quickly as possible, so:
  - a) Only those tasks directly relevant to that goal for a significant percentage of users are included in the standard installation experience.
  - b) Task performance must be measured and optimized appropriately.
2. Any configuration tasks the user performs during installation must also be available after the system is installed.
3. Installation policies will favor and encourage Solaris best practices for installing and managing systems.
4. Installation will leverage the best general-purpose technologies Solaris has available, we will avoid creating installation-specific mechanisms whenever possible.

The following sections describe various aspects of Caiman installation.

---

<sup>7</sup> [http://installzone.sfbay/projects/futures/solaris\\_install\\_work\\_list\\_2005.sxc](http://installzone.sfbay/projects/futures/solaris_install_work_list_2005.sxc)

<sup>8</sup> Caiman is inspired by the Red Hat Linux installer, which is known as Anaconda. Caimans and anacondas live in the same habitat, compete for the same food, and occasionally predate on each other.

## 4.1 Caiman User Experience

At the core of the Caiman user experience is a "live DVD" model, allowing the user to experiment with Solaris without committing to an installation at all, while also providing a simple path to installing Solaris. There are other possible uses for a live DVD that can be supported, depending on space availability and development resources; these will be discussed later. We will also discuss the network installation experience in a later section.

It's important to note at this point that the Solaris 2.x install media has always been a minimal live CD (now DVD). The system boots from the media the actual version of Solaris which will be installed and automatically executes the Solaris installation program. However, the environment presented is a subset of a normal Solaris installation, known as the "miniroot", and it provides only a limited X Windows environment that has no similarity to the default Solaris environment. The concept for Caiman replaces this limited environment with a fully-functional Solaris desktop environment that can be used comfortably by existing Solaris users and introduces new Solaris users to the system they'd see once they complete an installation.

Once the Solaris media has booted, the user will be presented with an attractive user interface which presents "Try" or "Install" choices. The "Try" choice dismisses the installer and provides a Solaris desktop, presenting a display which introduces the user to key elements of the Solaris desktop (the desktop will offer a simple, easily findable means to restart the installer later). Other options which might be presented at this point include "media verification", which would provide a basic integrity check that the install media is whole and uncorrupted, and "hardware compatibility test", which would run a series of tests and present an easily understandable report on whether the various devices on the system are supported by Solaris. When the system is able to connect to a network during this task, we would offer the option of downloading appropriate additional drivers as needed, integrating them into the installation environment, and automatically include them in the software to be installed on the system.

It's important at this point to note that the above-described user experience is not confined strictly to a workflow that requires the user to obtain or create physical media such as a DVD; as with the current installer, a network-based boot can deliver the same experience. We will explore implementing a Sun-provided WAN installation service, such that the user need only download at most a small stub boot system that can download and run the Caiman environment directly from Sun (or any other provider of an equivalent distribution).

In attempting to simplify the environment in which a live installer would need to be implemented, we will investigate the possibility of using Xen virtual machines to allow the installer to always run within the current release of the operating system rather than incurring the current requirement to implement to the oldest upgradeable release of Solaris. This would be a relatively long-term plan, depending on the versions of Solaris which Xen will support, and the strategy for virtualization on SPARC. Relatedly, we may also investigate providing a completely contained Caiman environment directly executable in an emulator such as QEMU. This would allow the live environment and installer to run within an existing installation of a non-Solaris operating system.

We have already done some initial design work on the user experience described here. David-John Burrowes of the xDesign team has implemented a simple Macromedia Flash prototype<sup>9</sup> in order to explore and illustrate some of the basic concepts, and conducted a small usability study<sup>10</sup> using it. Generally, the prototype was well-received, with those users experienced with the current Solaris installer feeling the prototype was a significant improvement.

#### **4.1.1 Initial Installation**

Caiman's installation user interface is focused on installing a usable Solaris as quickly and simply as possible, providing a primary path that requires a minimal level of user interaction for common installations. This suggests that the installation program will provide access to only the customizations which are absolutely necessary to perform an installation and achieve a usable system, with more complete configuration deferred to the post-installation boot of the installed system.

By default, the entire Solaris release will be installed; an advanced selection path will allow the user the option to customize the software selected for installation by functional groupings, with the ability to drill down to a package level, though limitations will be imposed to ensure that a supportable configuration is produced. The existing, high-level metaclusters will be removed as installation options. We do require that any software dependencies will be automatically resolved by the installer, so that a self-consistent installation is achieved. A path to choose installation from other media sources, such as an image on the network, will be available as well.

In addition to selecting the software to be installed, the user must also select where to install the software on the system's disks. Caiman supports coexistence with other operating systems on the same system, because this will often be the case with the developers we would like to capture and for whom this installation experience is most directly designed. Seamless coexistence requires the integration of functionality to easily manage the disk partitioning, including non-destructive resizing of FDISK partitions on x86 systems. Caiman will offer to install Solaris into any free space identified on a bootable disk device, but if no free space is identified, the user will be asked to instruct Caiman as to which space may be reused, or which occupied space should be compressed in order to obtain the needed free space.

Unlike the existing Solaris installer, the user will not be asked to select slices and mount points for a more detailed filesystem layout. This is because the default installation will use ZFS pools and filesystems to construct the basic filesystem layout. Advanced user options will allow the continued use of UFS as the installation filesystem for those customers who require it, but our direction is to de-emphasize this legacy technology. The usage of ZFS as the default filesystem simplifies our ability to offer safe patching and upgrades with rollback, and will allow for faster provisioning of zones.

Once disk space and software selection are completed, installation of Solaris will occur, with appropriate feedback displayed to the user as the installation progresses.

---

<sup>9</sup> <http://blogs.sun.com/roller/resources/dminer/prototype1.swf>

<sup>10</sup> [http://www.opensolaris.org/os/community/install/prototype\\_usability\\_study1/](http://www.opensolaris.org/os/community/install/prototype_usability_study1/)

### 4.1.2 System Configuration

Once software installation completes, the system will reboot to start the new operating system. In the case of an installation from either media or the network, in which the new version of the operating system is already booted, it may be possible to directly transition to running the newly installed software without a reboot of the hardware. This requires some investigation, but its attraction is that it would save a great deal of time on platforms where there is a long reset time for the firmware. Even if this idea proves infeasible, our goal is to remove any architectural requirement to reboot in order to effect system configuration changes so that on preinstalled systems, no reboot will be required.

After the newly-installed software has been started, system configuration will occur. Presently, this process is implemented by the `sysidtool` programs. However, the Network Automagic project<sup>11</sup> is re-architecting the Solaris network configuration model, its goal being to provide a more automated and flexible configuration experience which reflects modern network technology. Also, SMF and Visual Panels<sup>12</sup> offer the promise of well-integrated configuration tools and mechanisms which are not specific to installation. As a result, Caiman will replace the `sysidtool` programs with a new architecture which takes advantage of the progress made by these other projects.

We expect that many of the configuration settings will be automatically determined in most environments, and will redesign the user experience of system configuration around that assumption. Our goal, as noted above, is to require as little input as possible to achieve a usable system, and to work with other teams to provide the necessary system management tools so that settings can be modified as needed after initial installation and configuration have been completed. The conceptual design in the existing xDesign mockup reflects some of these goals, but will require additional modification once the Network Automagic model is more completely defined. One significant requirement to be addressed here is the need to eliminate reboots to effect some configuration changes.

### 4.1.3 Upgrade Installation

The user experience for upgrade installation will be essentially the same as that for initial installation. It will not be necessary to boot from Solaris media to perform such an upgrade – the existing, installed version of Solaris will be capable of running the installer and performing the upgrade to a copy of the system software while the system remains running, much as is the case with the current Live Upgrade technology. The option to boot from either media or the network and upgrade will continue to be supported, though the advantages of upgrading while “live” are compelling and should, over time, attract most customers.

In either upgrade scenario, the Caiman user interface will present the option to upgrade any supported Solaris installation found to be present on the system. Since this action is fundamentally selecting where to install the software, the upgrade selection is made at that stage of the installation user interface. An advanced mode will allow the user to customize the software selection for upgrade, with the standard path upgrading exactly the software

---

11 <http://www.opensolaris.org/os/project/nwam/>

12 <http://www.opensolaris.org/os/project/vpanels/>

that was previously installed, and additionally installing any new features.

#### **4.1.4 Automated Installation**

Caiman will continue to support a completely hands-free installation of Solaris, providing compatibility with existing environments that use Jumpstart for automated provisioning. As discussed previously in System Configuration on page 23, the existing `sysidtool` configuration utilities will be replaced; as a result, automated configuration of system parameters and services will migrate to using SMF features, so that the capability to apply a configuration “profile” will not be a specialized installation-time feature. We will provide migration tools for the `sysidcfg` mechanism currently used to automate the configuration process.

Caiman will improve the basic tools used to provision a Jumpstart environment by replacing the existing `add_install_client` and `setup_install_server` scripts with new tools which integrate with the Solaris administration toolset, and include tools to simplify setup for WAN installations. Discussion with the N1 provisioning product teams is required in order to more completely define the Caiman roadmap in this area, as it's undesirable to invest in duplicative efforts.

To aid in our efforts at simplifying automation, the interactive installer will automatically create and save a record of the installation selections to allow the installation to be duplicated later. We will also allow running the installer in a “dry-run” mode to generate an installation record without actually performing an installation.

#### **4.1.5 Replication and Recovery**

Conceptually, replication and recovery are two separate but similar tasks which Caiman will support.

First, Caiman will expand upon the existing Flash foundation to provide tools for creating a recoverable image of an installed system. Beyond merely generating an archive copy of the installed system, the tools will also include the ability to generate a completely bootable, installable ISO image, which can then be either transferred to media or merely stored on a server for later use over the network.

Second, Caiman will provide tools to generate a custom, installable distribution based on a repository of Solaris packages. These tools will be useful to customers who wish to define a self-contained “standard load”, as well as to developers who wish to create their own OpenSolaris-based distributions and possibly creators of OpenSolaris-based appliances.

#### **4.1.6 Diskless Systems**

With Caiman, we propose to replace the existing Solaris diskless NFS client support with a RAM disk model technologically similar to the current WANboot technology. The advantage of this model is that it can provide better administrative and operational scalability for true grid-style computing, which appears to be the principal market for this technology going forward. Validation of this proposal with current diskless customers is necessary, however.

### **4.1.7 Software Management**

Once the Solaris system is installed and running, inevitably the user will need additional software, or updates to existing installed software. Caiman will provide a model for repositories of software packages, tools to create and manage software package repositories, and end-user tools, both command-line and GUI, to automatically find, install, and update software from such repositories. The repositories will be authenticated, and packages within them may be signed for further integrity protection. The tools will make the task of removing software packages as easy as installing them, and the tools will also leverage system functionality such as ZFS snapshots in order to easily revert the system to a prior known state, allowing for an error-free backout of software whose installation proves undesirable.

Implementing repository support as described would allow the default Solaris installation to potentially be a smaller set of software than the entire release, as it would be simple enough for users to add software after installation that its initial absence would be less of a burden.

Management of the software on alternate boot environments is a topic for further research. One idea we wish to explore is installing each boot environment in such a way that it may also be booted as a virtual machine instance. This would eliminate the need for specialized options to commands and provide a logically consistent, and simple, model for administration of the system. The devil here is in the details, but conceptually it is very attractive.

### **4.1.8 Performance**

Solaris installation performance can be calculated as the aggregate of three steps:

- 1) the amount of time required for the user to download and construct the installation media
- 2) the amount of time required for a user to initiate the installation process and provide the necessary input to perform both standard and complex installations
- 3) the amount of time required to reach an installed, running system after all user input has been collected

To improve step 1, we could implement a Sun-managed WAN installation service, allowing for a smaller required download prior to beginning installation. This would probably elongate step 3, though if less than a full installation is required, time would be saved overall. Improvements here will require cooperation between Caiman and other Sun business units to achieve the desired results.

Improvement to step 2 should be a byproduct of the improved interactive interface and better-integrated tools for automated deployment described earlier, though this is not a given. The projects which implement those functions will be expected to baseline the performance of these tasks under the current Solaris installer, as well as similar tasks on competitive platforms, and set appropriate performance goals.

Our goal for step 3 must be to achieve near full utilization of the system's performance during this portion of the process; we should be limited only by the I/O rate of the system between the installation media and the storage devices used for the installed software.

Eliminating unnecessary reboots is also required to achieve optimal results on this portion of installation.

## 4.2 Community

As evidenced by the Blastwave and SunFreeware sites, the new OpenSolaris-based distributions which have sprung up, and numerous discussions in the OpenSolaris community, there is a strong interest in packaging and installation technologies. Accordingly, we have recently created the Installation and Packaging community<sup>13</sup> for OpenSolaris and released the SVR4 packaging tools project<sup>14</sup> to the community. The remainder of the existing installation code will be released to the community later in 2006.

We are hopeful of significant community participation in defining, designing, and implementing Caiman, and expect that it will be usable by other OpenSolaris-based distributions.

## 4.3 Caiman Project Roadmap

The Caiman user experience described in the previous sections represents a large, multi-year implementation effort. This section describes a possible breakdown of projects which would achieve that experience, along with dependencies and candidate technologies. We have not estimated resource requirements or schedules for the various projects, as most require technology investigation and selection, the results of which would have significant effects on the resources required and the resultant schedule.

### 4.3.1 Live CD

The Live CD project is responsible for constructing the tools needed to easily build live CD's for an OpenSolaris distribution, and constructing a live CD “product” for Solaris to use as a marketing tool. The live CD would include a current Solaris desktop environment, as well as hardware compatibility reporting. This project is not expected to provide a general installation capability, though it may provide a simple installation which would copy the CD contents to the system's disks. This project has no currently identified dependencies.

### 4.3.2 System Configurator

The System Configurator project is responsible for replacement of the `sysidtool` functionality, implementing a new set of configuration user interfaces and automation technology based on SMF, as described in System Configuration. The result of this project will improve the preinstalled system experience as well. This project intersects with, and likely has dependencies on, the Visual Panels and Network Automagic projects.

### 4.3.3 Disk Partitioner

The Disk Partitioner project is responsible for integrating tools capable of creating space for Solaris in a “brownfield” environment occupied by Windows, Linux distributions, or older

---

<sup>13</sup> <http://www.opensolaris.org/os/community/install/>

<sup>14</sup> [http://www.opensolaris.org/os/project/svr4\\_packaging/](http://www.opensolaris.org/os/project/svr4_packaging/)

Solaris versions. The GParted project<sup>15</sup> is a likely basis for this project. There are no identified dependencies for this project.

#### **4.3.4 Interactive Installer**

The Interactive Installer project is responsible for investigating options for an installation engine and user interface toolkit, selection of same, and implementing the interactive installation experience for initial installation and upgrade described in Initial Installation on page 22 and Upgrade Installation on page 23. The Purple Haze technology is a likely basis for this project. This project depends upon the Live CD, System Configurator and Disk Partitioner projects. Completion of this project would be equivalent to the scope of the concept car demo already produced by xDesign.

#### **4.3.5 Automation and Replication**

The Automation and Replication project is responsible for providing the installation automation and replication technologies described under Automated Installation and Replication and Recovery on page 24. It is also responsible for providing support for diskless systems as described under Diskless Systems on page 24. This project depends upon the Interactive Installer project for selection of an installation engine, and the System Configurator project for the configuration portions of automation.

#### **4.3.6 Software Manager**

The Software Manager is responsible for providing the software management tools and software repository services described under Software Management on page 25. It would also implement a Sun-supported master repository usable by those tools. The master repository would also be capable of supporting installation and upgrade over the Internet using the WAN installation technology. This project is dependent upon the completion of the Packaging and Patching strategy, the recommendations of which will likely determine the technologies used for this project.

---

<sup>15</sup> <http://gparted.sourceforge.net/>

## Acknowledgments

A number of people have contributed ideas for Solaris installation and review opinions on this paper. In no particular order, particular thanks to:

David-John Burrowes

Jeff McMeekin

Victor Nelson

Bev Crair

Mike Shapiro

Jeff Bonwick

Kevin Clarke

Jim Carlson

Geoff Arnold

Brian Wong

Jerry Jelinek

Lori Alt

and of course, the Solaris install team – Mary Ding, Gary Gere, Vassili Igouchkine, Tim Knitter, Glenn Lagasse, Ethan Quach, Sue Sohn, Sundar Yamunachari.