

AI Design 1.1 (Delta)

1.0 Introduction

This document describes the proposed changes to the Automated Installer for the AI spring release. Some of the changes are new features or enhancements to existing features. Some of them are major bugs that affect usability or quality. Once everything is finalized, these changes will go back to the design document and updated as version 1.1. Keeping it as a separate document makes it easier to navigate and concentrate on the changes. For reference, the AI design document version 1.0 is available at http://www.opensolaris.org/os/project/caiman/auto_install/ai_design_doc_v1.0a.pdf or you can pick up the document from the project document gate at hg.opensolaris.org/hg/caiman/caiman-docs/AI/ai_design_doc.odt.

1.1 Make Install services as SMF services

Currently, When an install service is created, there are following things happen on the install server:

- Enable Multicast DNS service, if needed
- Enable DHCP service, if needed
- Enable TFTP service if needed
- Start Apache web server on the special port for AI using the special configuration file
- Start AI web server for each service
- Register the named install service with the DNS service discovery. This starts a process and it should be controlled by the SMF service.

When the user issues 'installadm create-service', the code checks whether the needed SMF services like Multicast DNS (svc:/network/dns/multicast:default), DHCP service (svc:/network/dhcp-server:default), and TFTP service (svc:/network/tftp/udp6:default). These SMF services should be started automatically. That can be done by creating a SMF installadm service and making these services as dependent services.

The Apache web server serves the AI images for now. We need to create a SMF service for the Apache server and make it a dependant service for the SMF installadm service.

The AI web server and service registration process should be controlled by the SMF installadm service. To facilitate that all the information about a service is saved in a text file called 'service_data'. This file will be in the following format:

```
service_name:AI_directory:boot_file:txt_record:status
```

where service_name is the name of the created install service (provided as an argument for install_service or derived by the code), AI_directory is the target directory given as the argument for create-service,

boot_file is the name of the file (provided as an argument for install_service or derived by the code),

txt_record is the information recorded in the service and will be used by the client to contact the AI web server, and status indicates whether the install service is running or stopped by the user.

The installadm SMF service will use the 'installadm start' and 'installadm stop' to manage the AI web server and DNS service discovery registration process.

This task is covered by the bug 4488.

Right now, each install service starts its own AI web server on a different port. This is not the preferred approach. We should start only one AI web server and it should handle all the install services. That way, it is easier to manage install services and AI web service using SMF. Further more, we are using Apache web server for serving AI images to the client and AI web server for serving the manifests. These two should be consolidated and there should be only one web server for Automated Installer. This work will not be done by the next AI release but will be a high priority going forward.

1.2 DNS Unicast support

The AI preview release supported DNS multicast for install services. The DNS multicast only works within a subnet. Multicast DNS setup requires very little work and it works great if the install server and the clients are in the same subnet. Unfortunately they don't work across subnets. So we need unicast DNS setup to support installing when the install server and AI client are in different subnets. The Multicast DNS and unicast DNS are discussed in section 4.7 of AI design document V1.0. We can approach setting up unicast DNS in two different ways

- a) Enable dynamic update – Allow updating DNS records with the install service information when create-service is successful. This means either the user is running a local DNS server or the administrator for the DNS server enabled dynamic update. Even if the dynamic update is enabled, there should be a secure method to validate the client before allowing update. This is done for host and IP address update by some customers.
- b) Print out what needs to be added to the DNS server if the user selected unicast option with create-service. Also modify the create-service command so that the service name is mandatory for DNS unicast setup.

For the AI spring release, we will implement second option since it is less intrusive. We can handle the DNS dynamic update once we have a clear understanding of the benefits of enabling DNS dynamic update. If the service name is missing in the create-service command, we should not allow unicast setup. The user could setup some static DNS entries with known set of service names before setting up services using create-service. We are not going to verify whether the user has setup DNS entries for the service in the create-service. That could be done separately by different tools.

If DNS configuration is not possible and multicast DNS is not available then we could use installation configuration files menu.lst (X86) and install.conf (SPARC) to setup the Automated Installer. Currently we are adding only the service names in those configuration files. We could add the information we register (text record) with DNS service in the configuration files. The client could first check for unicast then multicast. If there is no service information available, it will use the information in the configuration files. This will allow booting and installing across subnets if unicast DNS is not setup.

The create-service has an option -u to setup unicast DNS. If -u option is specified then create-service expects the -n <install_service_name> as follows:

```
# installadm create-service [-u host:port] -n <service_name> -s <AI_source_image>
```

<target_directory>

The DNS record for an install service will look like:

1. Domain Enumeration

```
;  
; These records invite clients to browse this domain.  
; (The '@' sign means "the current domain" --  
; i.e. the domain being described by this zone file.)  
  
b._dns-sd._udp IN PTR @ ; b = browse domain  
lb._dns-sd._udp IN PTR @ ; lb = legacy browse domain  
; (domain that will be used for clients that don't specify a domain to browse)
```

2. List of named service entities

```
;  
; These records list the named 'OSInstall' service entities we want clients  
; to be able to discover. You can use upper-case, punctuation,  
; UTF-8 rich text, etc., in service names, but certain punctuation  
; like spaces and dots needs to be escaped with a backslash in the  
; name server zone file (on the wire it's sent as raw data, not escaped).
```

```
_OSInstall._tcp PTR _test_service._OSInstall._tcp
```

3. SRV & TXT records describing each service entity named above

```
;  
; These records tell clients how to access the named 'OSInstall' service entities  
; listed above. In the example below, fill in the correct host name.  
; If it's a fully qualified host name, it needs to end with a dot, as shown.  
; If it's a partial host name relative to the current zone (e.g. just "testdomain",  
; with the remainder of the name being implicit) then don't end it with a dot.
```

```
_default_OSInstall._tcp SRV 0 0 10000 testdomain.sun.com.  
TXT aiwebserver=testdomain:10000
```

1.3 Streamline AI schema and AI manifest related with Install target

With the 2008.11 preview release, the install target could be defined and the partitions can be created with size specified in sectors. This is not very useful for lot of users. We are expanding the partition and slice operations considerably for the AI spring release. We are planning to implement the following enhancements.

- Flexibility of size units (MB/GB/TB/sectors) for partition and slices
- Create/delete/preserve VTOC slices
- Create/delete fdisk partitions
- Allow creating non-solaris partitions (Diagnostic, Linux, FAT16, FAT32 and any other simple partitions)
- Install on to existing slices (remove the restriction of installing to slice 0)
- Disk selection based on parameters like size, vendor etc.
- Customize the software to be installed (Add/delete packages)

- Host name, Keyboard and locale as configured from SC manifest

These enhancements are already filed as bugs and some of them are already implemented. The latest AI schema and examples are available at section 2 of this document.

1.4 Improve the AI manifest usability

AI has 3 types of manifests called AI manifest, SC manifest and criteria. The manifests are discussed in the [AI manifest](#) document. It also has some examples.

1.4.1 Restructure criteria portion in a composite manifest to allow for a null criteria.

In the 2008.11 AI preview release, criteria manifest included AI manifest and SC manifest as follows:

```
<ai_criteria_manifest>
  <ai_criteria name="...">
    .....
  </ai_criteria>
  <ai_manifest_file URI="./ai_manifest1.xml"/>
  <sc_manifest_file name="AI" URI="./sc_manifest1.xml"/>
</ai_criteria_manifest>
```

This means always the manifests are published with a service with a criteria. In some cases, the user may want to create manifests and publishes for testing. Later on the criteria to match clients with manifests can be added. To facilitate allowing NULL criteria, the format of the composite manifest will be changed as follows:

```
<ai_composite_manifest>
  <optional>
    <ai_criteria name="...">
      .....
    </ai_criteria>
  </optional>
  <ai_manifest_file URI="./ai_manifest1.xml"/>
  <sc_manifest_file name="AI" URI="./sc_manifest1.xml"/>
</ai_composite_manifest>
```

1.4.2 Adding or Changing criteria of a manifest

Since we are planning to allow manifests to be published with out a criteria, we need commands to add a criteria. We should also allow a criteria to be changed or deleted. The following additions will be made to installadm to support modifying criteria. The 2008.11 version of the installadm sub commands can be found in section 4.3 of the AI design document v1.0.

```
installadm set-criteria -n <svcname> -m <manifest_name>
criteria=<file_name>
```

The '*set-criteria*' sub command will add the new criteria or change the existing criteria of a manifest already added to the given service. When a client send a request with a matching criteria, the service will return the corresponding manifest. In the '*set-criteria*' usage, *svcname* is the name of the service, *manifest_name* is the name of the manifest you want to add the criteria and the *file_name* has the

criteria information. If the manifest that is given as the input already has a criteria, it will be overwritten with the new criteria.

```
installadm get-criteria -n <svcname> -m <manifest_name> [-o  
file_name]
```

The '*get-criteria*' sub command will retrieve the existing criteria of a manifest published with a service. This sub command can be used to retrieve the current criteria, make changes and set it as the new criteria of a manifest. In the *get-criteria* usage, *svcname* is the name of the service, and *manifest_name* is the name of the manifest whose criteria is requested. The *file_name* is an optional argument. If it is provided, the criteria will be saved in the *file_name*. Otherwise, it will be displayed on the screen.

```
installadm clear-criteria -n <svcname> -m <manifest_name>
```

The '*clear-criteria*' sub command will remove the existing criteria of a manifest published with a service. In the '*clear-criteria*' usage, *svcname* is the name of the service, and *manifest_name* is the name of the manifest whose criteria is to be removed.

1.4.3 Exporting manifests

When a manifest is published to a service, a file name that contains the manifest should be provided as an input. The service validates the manifest and copies the manifest to its internal format. The original file is not used any more. The changes made to the original file will not be reflected in the service unless it is published again. So the user might want to know what is currently published with a service. We are adding a sub command called '*export-manifest*' to retrieve a manifest from a service. The proposed syntax is as follows:

```
installadm export-manifest -n <svcname> -m <manifest_name> [-o  
file_name]
```

The '*export-manifest*' sub command will retrieve a manifest that was already published with a service. This sub command can be used to retrieve the current manifest, make changes and publish it back to the same service or a different service. In the '*export-manifest*' usage, *svcname* is the name of the service, and *manifest_name* is the name of the manifest whose contents are requested. The *file_name* is an optional argument. If it is provided, the manifest will be saved in the *file_name*. Otherwise, it will be displayed on the screen.

In the 2008.11 AI preview, the sub command to publish a manifest is '*add*' and the sub command to remove a manifest is '*remove*'. Check section 4.3 of the AI design document v1.0 for more information. To be consistent with the new sub command '*export-manifest*', we will be changing sub command '*add*' to '*add-manifest*' and '*remove*' to '*remove-manifest*'. The syntax of *add-manifest* and *remove-manifest* will be:

```
installadm add-manifest [-d] -n <svcname> -m <manifest_name>  
installadm remove-manifest -n <svcname> -m <manifest_name>
```

There is a new option *-d* added to the sub-command to denote that the manifest being added will be the default manifest for the service. See the section 1.4.5 for more information about the default manifests.

1.4.4 Default manifest specification

When a new install service is created using create-service sub command, a default AI manifest is copied to the service. This default manifest allows the user to perform OpenSolaris installation using AI with out setting up any special setup. Since the default manifest is generic, it may not meet everyone's requirements. The user can change the default manifest but it was not straight forward in the 2008.11 AI preview release. The user has to create a AI manifest file with the value of the name field set to 'default' and publish it using 'add' command. We are making changes to make it easier by introducing a property called 'default-manifest' to each service. It will be set to the default manifest shipped with the server tools package when a new service is created. The default could be changed to the user specified AI manifest by using the set property. The sub command set was discussed in the section 4.3 of the AI design document 1.0 but never got implemented in 2008.11 AI preview. There are two ways to change the default manifest of a service

```
installadm set -p default-manifest=<name> <svcname>
installadm add-manifest [-d] -n <svcname> -m <manifest_file>
```

1.4.5 AI Manifest versions

New features will be added to Automated Installer, which means the AI schema and AI manifest may going through some incompatible changes. We already have version for the AI schema. To validate a new manifest, it will be useful to have the version in the AI manifest. The user could add the version to the manifest but it should be taken care of the install server tools. When a manifest is published to a service, it is validated against the current schema. If it is successful, the server tools adds the version to the manifest if it is not there already before storing in the server database. When the user retrieves a manifest from the service next time, it will have a version.

1.5 Enhance Server tools reporting

The 2008.11 AI preview provides some support for getting information like list of install services running on the network and the list of manifests published with a service. The next AI release will build on that and provide more visibility on the services, manifests, criteria, AI images and clients setup on this install server. See section 4.3 of the AI design document 1.0 for more information about 2008.11 version of the 'list' sub command. The proposed new syntax for sub command 'list' is as follows:

```
installadm list [-a | -c | -m | -s | -C | -n <svcname> | -r]
```

All the arguments are optional to the sub command 'list'.

With out any options, the sub command list will display all the install service started from this install server.

If '-a' option is provided, all services, manifests, criteria, AI images and configured clients will be displayed.

If '-c' option is provided, only the clients configured on this server along with the image and service will be displayed.

If '-m' option is provided, all the manifests published on this server will be displayed

If '-s' option is provided, all the services running on this install server along with the published manifests will be displayed.

If '-C' option is provided, all the criteria information will be displayed.

If '-n <svcname>' is provided, manifests published on the service will be displayed.

If '-r' is provided, it will show all services published on the local subnet will be displayed. This is required because multicast DNS is broadcast based protocol. It is good to know what are the services a client in the subnet could see.

Once we start implementing the enhancements to list sub command, we may decide that some options may not be necessary. For example, when the user reuests for manifests, the criteria information can be displayed and vice-versa. Thus making one of the options redundant.

The format of the output is not yet decided. Once we have the information, this document will be updated to reflect the format.

1.6 Use standard format for Ethernet Address, and IP address

In 2008.11 AI preview release, the criteria for Ethernet address (MAC), Network address (NETWORK) and IP address (Ipv4) don't use the standard format. See section 4.9 and 4.7.5.2 of the AI design document V1.0 for more information. For example if the Ethernet address is 1:e0:c0:12:34:56, then MAC should be 01E0C0123456 (All upper case string with ':' removed). For the spring release it will accept the standard form (one with the ':' intact) as well as the string format supported by 2008.11 preview release.

The Criteria file is used by '*installadm add -m manifest -n service*'

Typical criteria file for using Ethernet address as a criteria:

```
<ai_criteria_manifest>
  <ai_criteria name="MAC">
    <range>
      01E0C0123456
      01E0C0123456
    </range>
  </ai_criteria>
  <ai_manifest_file URI="./ai_manifest1.xml"/>
  <sc_manifest_file name="AI" URI="./sc_manifest1.xml"/>
</ai_criteria_manifest>
```

The criteria file can changed as follows:

```
<ai_criteria_manifest>
  <ai_criteria name="MAC">
    <range>
      1:e0:c0:12:34:56
      1:e0:c0:12:34:56
    </range>
  </ai_criteria>
  <ai_manifest_file URI="./ai_manifest1.xml"/>
  <sc_manifest_file name="AI" URI="./sc_manifest1.xml"/>
</ai_criteria_manifest>
```

1.7 Quality Improvements

We will address all blockers and critical bugs till release candidate 1 of the AI next release. After that we will be careful in integrating bug fixes. We will try to address all P1 and P2 bugs for the next AI release. The rest of the bugs will be handled case by case basis. To get the current list of open AI bugs, please go to

http://defect.opensolaris.org/bz/buglist.cgi?cmdtype=dorem&reanction=run&namedcmd=AI_bugs&sharer_id=219

2 AI schema, AI manifest and usage examples

The target device definition for the 2008.11 AI preview release was simple. It allowed to specify a device name, create partition and slices using only sectors. The enhancements to the target device are discussed below

2.1 Target device enhancements

Target Device : Zero or more elements can be grouped together

<i>Element name</i>	<i>Explanation</i>	<i>Values</i>
target_device_name	Device Name cXtXdX format or World Wide Name format like cXtX....dX	String like c0t0d0 or c0t2000002037CD9F72d0
target_device_size	size of partition – see partition_size_units, default MB	integer
target_device_vendor	targets only disks of a given vendor	text, case ignored - Hitachi, EMC, Seagate, Maxtor, Sun – use iostat (1M) -E to see vendor spelling
target_device_type	driver class of target device	ATA, SCSI, FC, SAS, USB, others
target_device_use_solaris_partition	targets Solaris2 partitions already created	true/false
target_device_install_slice_number	number of slice to install Solaris – will be created automatically if it does not already exist	0,1,3,4,5,6, or 7, default 0
target_device_overwrite_root_zfs_pool	targets existing zfs root pool, overwriting it	true/false

Partitioning :

Supported operations (element: partition_action): create and delete.

Create partition: create a new fdisk partition on a disk (element: partition_action=create)

<i>Element name</i>	<i>Explanation</i>	<i>Attributes</i>	<i>Values</i>
partition_start_sector	fdisk partition starting sector	optional. If omitted, AI finds best fitting location	integer 0 - (#sectors on disk - 1)
partition_size	size of partition – see partition_size_units, default MB		integer number or “max_size”, for largest possible contiguous unused region on disk
partition_size_units	units for partition_size	optional, default to MB	MB, GB, TB, sectors (1 sector = 512 bytes) –

partition_type	specify partition type	some variations accepted integer – 191 for Solaris2
----------------	------------------------	--

Delete Partition: delete a partition from a disk (element: partition_action=delete)

<i>Element name</i>	<i>Explanation</i>	<i>Attributes</i>	<i>Values</i>
partition_start_sector	starting sector of partition to delete	optional	integer
partition_size	size of partition to delete in sectors	optional	integer
partition_number	partition number to delete		integer 1-4
partition_type	type of partition to delete, assuming only one partition of that type exists		integer, 191 for Solaris2

Disk Slices/VTOC

Supported operations (element: slice_action): create, delete, and preserve

Create slice: create a VTOC slice on a disk (element: slice_action=create):

<i>Element name</i>	<i>Explanation</i>	<i>Attributes</i>	<i>Values</i>
slice_number	number of VTOC slice ID to create	required for each slice to create	0 through 7 except 2
slice_size	size of slice, default MB, see slice_size_units		integer or “max_size” - use largest contiguous unused region in a fdisk partition or on a SPARC disk
slice_size_units	units expressed in slice_size	optional, default MB	MB, GB, TB, sectors (1 sector = 512 bytes) – some variations accepted

Delete Slice: delete a VTOC slice from a disk (element: slice_action=delete):

<i>Element name</i>	<i>Explanation</i>	<i>Attributes</i>	<i>Values</i>
slice_number	VTOC slice number to delete	required for each slice to delete	0 through 7 except 2

Preserve Slice: preserve a VTOC slice on a disk, deleting others

<i>Element name</i>	<i>Explanation</i>	<i>Attributes</i>	<i>Values</i>
slice_number	VTOC slice number to preserve – if one slice is preserved, all slices not preserved will be deleted	required for each slice to preserve	0 through 7 except 2

2.2 AI manifest Examples

The user wants to know how to add new features to the AI manifests. This section will provide some examples of the new features. For 2008.11 AI preview release, examples of manifests are found in the [AI manifest](#) document.

2.2.1 Installing on slice 4 of the target device c0t0d0

```
<ai_manifest name="install_on_slice_4">
  <ai_target_device>
    <target_device_name>c0t0d0</target_device_name>
    <target_device_install_slice_number>4</target_device_install_slice_n
umber>
  </ai_target_device>
  <ai_pkg_repo_default_authority>
    <main url="http://pkg.opensolaris.org" authname="opensolaris.org"/>
    <mirror url=""/>
  </ai_pkg_repo_default_authority>
  <ai_packages>
</ai_manifest>
```

2.2.2 Deleting existing FAT32 partition and creating Solaris2 partitioning

```
<ai_manifest name="create_delete_partitions">
  <ai_target_device>
    <target_device_name>c0t0d0</target_device_name>
  </ai_target_device>
  <ai_device_partitioning>
    <partition_action>delete</partition_action>
    <partition_number>3</partition_number>
    <partition_type>11</partition_type>
  </ai_device_partitioning>
  <ai_device_partitioning>
    <partition_action>create</partition_action>
    <partition_number>3</partition_number>
    <partition_size>30</partition_size>
    <partition_size_units>GB</partition_size_units>
    <partition_type>Solaris</partition_type>
  </ai_device_partitioning>
  <ai_pkg_repo_default_authority>
    <main url="http://pkg.opensolaris.org" authname="opensolaris.org"/>
    <mirror url=""/>
  </ai_pkg_repo_default_authority>
  <ai_packages>
</ai_manifest>
```

2.2.3 Delete slice 0 and create slice 0

```
<ai_manifest name="create_delete_slices">
  <ai_target_device>
    <target_device_name>c0t0d0</target_device_name>
    <target_device_install_slice_number>0</target_device_install_slice_n
umber>
  </ai_target_device>
  <ai_device_vtoc_slices>
    <slice_action>delete</slice_action>
    <slice_number>0</slice_number>
  </ai_device_vtoc_slices>
  <ai_device_vtoc_slices>
    <slice_action>create</slice_action>
    <slice_number>0</slice_number>
    <slice_size>20</slice_size>
    <slice_size_units>GB</slice_size_units>
  </ai_device_vtoc_slices>
  <ai_pkg_repo_default_authority>
    <main url="http://pkg.opensolaris.org" authname="opensolaris.org"/>
    <mirror url=""/>
  </ai_pkg_repo_default_authority>
  <ai_packages>
</ai_manifest>
```

2.2.4 Preserve slice 7 and create slice 3

```
<ai_manifest name="create_preserve_slices">
  <ai_target_device>
    <target_device_name>c0t0d0</target_device_name>
    <target_device_install_slice_number>0</target_device_install_slice_n
umber>
  </ai_target_device>
  <ai_device_vtoc_slices>
    <slice_action>preserve</slice_action>
    <slice_number>7</slice_number>
  </ai_device_vtoc_slices>
  <ai_device_vtoc_slices>
    <slice_action>create</slice_action>
    <slice_number>3</slice_number>
    <slice_size>20480</slice_size>
    <slice_size_units>MB</slice_size_units>
  </ai_device_vtoc_slices>
  <ai_pkg_repo_default_authority>
    <main url="http://pkg.opensolaris.org" authname="opensolaris.org"/>
    <mirror url=""/>
  </ai_pkg_repo_default_authority>
  <ai_packages>
</ai_manifest>
```

2.3 AI schema

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

=====
RelaxNG schema for Automatic Installer input manifest specification.

Contains schema rules and content specification for AI manifest.

```

=====
-->

<grammar
  xmlns="http://relaxng.org/ns/structure/1.0"
  datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">

  <start>
    <element name="ai_manifest">
      <attribute name="name"/>
      <!-- General automatic Installation parameters -->
      <ref name="auto_installation_manifest"/>
    </element>
  </start>

  <!--
  =====
  Parameters used for Automatic installation
  =====
  -->
  <define name="auto_installation_manifest">
    <interleave>
      <!-- Any element order is OK. -->
      <optional>
        <element name="ai_target_device">
          <ref name="ai_target_device_contents"/>
        </element>
      </optional>

      <zeroOrMore>
        <element name="ai_device_partitioning">
          <ref
name="ai_device_partitioning_contents"/>
        </element>
      </zeroOrMore>

      <zeroOrMore>
        <element name="ai_device_vtoc_slices">
          <ref name="ai_device_vtoc_slices_contents"/
>
        </element>
      </zeroOrMore>

      <optional>
        <element name="ai_device_zfs_root_pool">
          <ref name="ai_zfs_root_pool_contents"/>
        </element>
      </optional>

      <optional>
        <element name="ai_device_swap">
          <ref name="ai_device_swap_contents"/>
        </element>
      </optional>

      <optional>
        <element name="ai_pkg_repo_default_authority">
          <ref name="ai_pkg_repo_contents"/>
        </element>
      </optional>
    </interleave>
  </define>

```

```

        </element>
    </optional>

    <zeroOrMore>
        <element name="ai_pkg_repo_addl_authority">
            <ref name="ai_pkg_repo_contents"/>
        </element>
    </zeroOrMore>

    <zeroOrMore>
        <element name="ai_packages">
            <ref name="ai_package_contents"/>
        </element>
    </zeroOrMore>

    <optional>
        <element name="ai_http_proxy">
            <ref name="ai_http_proxy_contents"/>
        </element>
    </optional>

    </interleave>
</define>
<!--
=====
Selections for AI target Device specification
=====
-->
<define name="ai_target_device_contents">
    <interleave>
        <optional>
            <!-- device name like c0t0d0 or
                MPXIO name like c0t2000002037CD9F72d0 -->
            <element name="target_device_name">
                <text/>
            </element>
        </optional>
        <optional>
            <element name="target_device_type">
                <text/>
            </element>
        </optional>
        <optional>
            <element name="target_device_vendor">
                <text/>
            </element>
        </optional>
        <optional>
            <element name="target_device_size">
                <text/>
            </element>
        </optional>
        <optional>
            <element name="target_device_install_slice_number">
                <choice>
                    <value>0</value>
                    <value>1</value>
                    <value>3</value>
                    <value>4</value>
                </choice>
            </element>
        </optional>
    </interleave>
</define>

```

```

                <value>5</value>
                <value>6</value>
                <value>7</value>
            </choice>
        </element>
    </optional>
</optional>
    <element
name="target_device_use_solaris_partition">
        <data type="boolean"/>
    </element>
</optional>
</optional>
    <element
name="target_device_overwrite_root_zfs_pool">
        <data type="boolean"/>
    </element>
</optional>
</interleave>
</define>

<!--
=====
Selections for AI target device partitions specification
=====
-->
<define name="ai_device_partitioning_contents">
    <interleave>
        <element name="partition_action">
            <choice>
                <value>create</value>
                <value>delete</value>
            </choice>
        </element>
        <optional>
            <element name="partition_number">
                <data type="unsignedByte"/>
            </element>
        </optional>
        <optional>
            <element name="partition_start_sector">
                <data type="long"/>
            </element>
        </optional>
        <element name="partition_size">
            <data type="unsignedLong"/>
        </element>
        <element name="partition_type">
            <data type="unsignedByte"/>
        </element>
        <optional>
            <element name="partition_size_units">
                <ref name="disk_space_size_units"/>
            </element>
        </optional>
    </interleave>
</define>
<!--
=====

```

Selections for AI target device vtoc slices specification

```
=====
-->
<define name="ai_device_vtoc_slices_contents">
  <interleave>
    <element name="slice_action">
      <choice>
        <value>create</value>
        <value>delete</value>
        <value>preserve</value>
      </choice>
    </element>
    <element name="slice_number">
      <data type="unsignedByte"/>
    </element>
    <element name="slice_size">
      <data type="unsignedLong"/>
    </element>
    <optional>
      <element name="slice_size_units">
        <ref name="disk_space_size_units"/>
      </element>
    </optional>
  </interleave>
</define>
```

<!--

Selections for AI ZFS Root Pool specification

```
=====
-->
<define name="ai_zfs_root_pool_contents">
  <interleave>
    <optional>
      <element name="enable_mirrored_root">
        <data type="boolean"/>
      </element>
    </optional>
    <optional>
      <element name="zfs_options">
        <text/>
      </element>
    </optional>
  </interleave>
</define>
```

<!--

Selections for AI swap specification

The target device will be used as swap device

```
=====
-->
<define name="ai_device_swap_contents">
  <element name="ai_swap_size">
    <data type="unsignedLong"/>
  </element>
</define>
```

<!--

```
=====  
Define an authority and its mirror backups.  
=====
```

```
-->  
<define name="ai_pkg_repo_contents">  
  <element name="main">  
    <ref name="ai_repo_name"/>  
  </element>  
  <zeroOrMore>  
    <element name="mirror">  
      <ref name="ai_mirror_name"/>  
    </element>  
  </zeroOrMore>  
</define>  
  
<define name="ai_repo_name">  
  <attribute name="authname">  
    <text/>  
  </attribute>  
  <attribute name="url">  
    <text/>  
  </attribute>  
</define>  
  
<define name="ai_mirror_name">  
  <attribute name="url">  
    <text/>  
  </attribute>  
</define>
```

```
<!--
```

```
=====  
Define AI proxy  
=====
```

```
-->  
<define name="ai_http_proxy_contents">  
  <attribute name="url">  
    <text/>  
  </attribute>  
</define>
```

```
<!--
```

```
=====  
Define AI package and clusters  
=====
```

```
-->  
<define name="ai_package_contents">  
  <optional>  
    <element name="package_name">  
      <text/>  
    </element>  
  </optional>  
  <optional>  
    <element name="package_authority">  
      <text/>  
    </element>  
  </optional>  
  <optional>  
    <element name="package_version">
```

```

                <text/>
            </element>
        </optional>
    </optional>
    <optional>
        <element name="package_FMRI">
            <text/> <!-- http or ftp format -->
        </element>
    </optional>
</define>
<!--
=====
General definitions
=====
-->
<!--
=====
Units measuring sizable regions of space on disk drives
=====
-->
<define name="disk_space_size_units">
    <choice>
        <value>sectors</value>
        <value>secs</value> <!-- disk space sectors (512 bytes)
-->
        <value>sec</value> <!-- disk space sectors (512 bytes)
-->
        <value>s</value>
        <value>megabytes</value>
        <value>megabyte</value>
        <value>mb</value>
        <value>m</value>
        <value>gigabytes</value>
        <value>gigabyte</value>
        <value>gb</value>
        <value>g</value>
        <value>terabytes</value>
        <value>terabyte</value>
        <value>tb</value>
        <value>t</value>
        <value>SECTORS</value>
        <value>SECS</value> <!-- disk space sectors (512 bytes)
-->
        <value>SEC</value> <!-- disk space sectors (512 bytes)
-->
        <value>S</value>
        <value>MEGABYTES</value>
        <value>MEGABYTE</value>
        <value>MB</value>
        <value>M</value>
        <value>GIGABYTES</value>
        <value>GIGABYTE</value>
        <value>GB</value>
        <value>G</value>
        <value>TERABYTES</value>
        <value>TERABYTE</value>
        <value>TB</value>
        <value>T</value>
    </choice>
</define>

```

</grammar>