IBM VisualAge Smalltalk Enterprise Version 4.5a Fixpack

Contents

Chapter 1. Fixpack Files and Installation

VisualAge Smalltalk Enterprise Fixpack 4.5a is a maintenance release to fix problems reported against VisualAge for Smalltalk Version 4.5. Fixpack 4.5a also includes some product improvements. This document contains installation information, product fix descriptions, and product improvement highlights.

The fixpack is packaged in multiple files according to installable components. For example, there is a fixpack file for the base feature, a fixpack file for the Server Workbench feature, a fixpack file for the ObjectExtender feature, and so on. The fixpack files are further packaged according to the target operating system. The fixpack files will overlay files in the file directory tree structure where your current VisualAge installation resides.

If you originally installed:	Then you should apply fixpack:
VisualAge Smalltalk Enterprise Client	b_v45a_?.[zip | tar.Z]
VisualAge Smalltalk Enterprise - Manager Library	m_v45a_?.[zip | tar.Z]
VisualAge Smalltalk Enterprise - Stand Alone	b_v45a_?.[zip | tar.Z]
ObjectExtender	o_v45a_?.[zip | tar.Z]
Server Runtime	s_v45a_?.[zip | tar.Z]
Server Workbench	w_v45a_?.[zip | tar.Z]
UML Designer	u_v45a_?.[zip | tar.Z]

```
        where ? is:
            a - AIX
            o - OS/2
            w - Windows 3.11, 95, NT
            h - HP_UX
            s - Solaris

     Also note that fixpack files for AIX, HP_UX, and Solaris are not
     packaged as zip files but as compressed,  tar files.
```

Before installing the fixpack, ensure that you have the correct version of the product installed that this fixpack supports. The product version is displayed on the logo when you start VisualAge. The product version is also found in the opening text on the System Transcript. If you are unable to locate the version information in the System Transcript, you can redisplay the product version from the VisualAge Organizer by pointing to ″Help″ and selecting ″About this product″.

Installation instructions

Before installing the fixpack

1. Before beginning the fixpack installation, as a precaution, you should back up your VisualAge installation.
2. Take note of any features that you have currently installed because you will need to re-load them once the fixpack files are installed.

Installing the fixpack

1. Download the fixpack files you need to the subdirectory where your VisualAge installation resides.
   - For AIX, HP-UX, and Solaris, the installation directory is /opt/IBMvast.

   - For OS/2 and Windows, this is the directory that you specified when you installed VisualAge.

     Remember that VisualAge Server was probably installed in a different directory than VisualAge Client or VisualAge Stand Alone.

2. Unzip (or uncompress and untar) the fixpack files. On OS/2 and Windows, ensure that you unzip the fixpack using whatever option your unzip program provides that will preserve the subdirectory structure of the fixpack file. See example commands below.

   If you unzip (or untar) more than one fixpack file, you should unzip (or untar) the base file (b_v45a*) first.

   Warning: The fixpack files assume the directory structure as originally created by the VisualAge Version 4.5 install program. If you have rearranged this structure, then the fixpack may not install correctly.

   ```
   Example commands:

   unzip -d b_v45a_w.zip (OS/2 or Windows)

   zcat b_v45a_a.tar | tar -xvf -
   ```

3. If you originally did a VisualAge Client installation, then copy he fixpack DAT files (45a_*.dat) from the import subdirectory on your machine to the import directory on the Manager Server.
4. Start your image. Once the image has finished loading, From the System Transcript menu bar, select Tools, then Load/Unload Features.
5. Load the ″VisualAge: VisualAge Base″ feature and any other features that were loaded before you unzipped the fixpack files.
6. The fixpack supplies both new and replacement files (for example, the Version 4.5 abttvv40.ctl will be overwritten with a newer version by the fixpack). Therefore, if you reinstall a Version 4.5 feature, you must also reinstall the corresponding fixpack file.

If you are the library administrator for your team, you may want to baseline this image and distribute it to your clients as a new starting point for their subsequent development work.

Chapter 2. Component updates

The following components have been updated and improved. Details for each are given in this section.

- EMSRV
- MQSeries
- Server Smalltalk (SST)
- ObjectExtender
- UML Designer
- Performance improvements for ScaledDecimal

EMSRV Update

This fixpack contains EMSRV 6.23 which is a service refresh.

If you are running EMSRV as a Windows NT service, you should reinstall the service (emsrv -install If you are running EMSRV as a Windows NT service, you should reinstall the service (emsrv -install [optional parameters]) after installing the fixpack.

If you are using EMSRV for Netware, you will find replacement zip files in the netware subdirectory. These should be unzipped and installed following the same procedure that was used to install your EMSRV 6.22 support from the VisualAge Smalltalk Enterprise 4.5 CD.

MQSeries Version 5 Support

A number of improvements have been made to support MQSeries Version 5.

Distribution list feature

In order to support the distribution list feature added to version 5 of MQ, we have added the class AbtMQDistributionList. You use this class much like you use the AbtMQqueue class. In fact, the majority of methods in these classes are identical. We have factored out all common behavior into the class AbtMQAbstractQueue; both AbtMQqueue and AbtMQDistributionList inherit from AbtMQAbstractQueue.

AbtMQDistributionList maintains an OrderedCollection of instances of AbtMQDistributionListQueueInfo; each member of the OrderedCollection corresponds to one destination for the distribution list. You must initialize each instance of AbtMQDistributionListQueueInfo with the queue name and information about the queue manager (which you can initialize either by passing the name of the queue manager or by passing an instance of AbtMQqm). The instances may optionally contain any information that would normally go into an MQPMR structure.

You may build the OrderedCollection in AbtMQDistributionList in one of two ways. You could build each instance of AbtMQDistributionListQueueInfo yourself, passing the instances to the AbtMQDistributionList through the method addDestination:. Alternatively, you could use the method AbtMQDistributionList >> addQueue:queueManager: and allow this method to automatically build the AbtMQDistributionList instances for you.

We have added the classes AbtMQORStruct and AbtMQPMRStruct to support the new structures for MQOR and MQPMR. We have also updated the classes AbtMQPMOStruct and AbtMQODStruct to contain the optional fields used with distribution lists. Note that if you are using any of the fields in the MQPMR structure, you must use the method AbtMQDistributionList >> pmoPutMsgRecFields: to set the MQPMO structure's putMsgRecFields value.

With distribution lists, it is now possible to get multiple errors back from one command. We have added the class AbtMQMultipleErrors (which inherits from AbtMQError) to handle these situations. If your call results in multiple errors, you will receive an instance of AbtMQMultipleErrors where you would normally receive an AbtMQError.

You can also take advantage of this new feature if you are programming visually. The settings notebooks for the parts for the MQ Series Connection Spec and MQ Series Proc Dialog have been updated. On the ″Destination″ tab for each of these parts, there is now a checkbox labelled ″Use Distribution List″ next to the request queue name and reply queue name. Since MQ does not allow you to use distribution lists on get calls, you may mandate the use of a distribution list on either the request queue or the reply queue, but not both. If this part is acting as a server, you can use the reply queue as a distribution list; otherwise, you can use the request queue as a distribution list. If you specify one of the queues as a distribution list, you can build the list for this queue on the ″Distribution List″ notebook page.

Enhanced message grouping

The message grouping feature in MQ version 5 is accessed through new fields in the MQGMO and MQMD structures. We have updated the classes AbtMQGMOStruct and AbtMQMDStruct to contain these new fields. We have also added methods to AbtMQMessage that allow access to the new fields in MQMD structures.

Queue manager coordination of global units of work

MQ version 5 has added the MQBEGIN verb in order to allow the queue manager to coordinate global units of work. You may call this new verb through the method AbtMQqm >> begin.

Effect on MQ version 2 users

Every new field contained in an MQ structure that is new for MQ version 5 is strictly optional. Therefore, users of MQ Series version 2 can ignore these new fields and still use the fixpack. If one of these new, optional fields is set, the MQ version 2 queue manager will ignore the value. MQ version 2 users should avoid using the AbtMQDistributionList class. No action is necessary for migration of MQ apps between 4.5 and 4.5a.

Server Smalltalk Update

This fixpack contains a service refresh of Server Smalltalk (SST).

Summary of SST changes

A summary of changes for this refresh are as follows.

- New API which simplifies the creation and naming of SST infrastructure for frequent use-cases, including creating applications contexts for IIOP, RMI and By-Reference/Logical-Process Method Invocation. (See SstApplicationContext class methods #create*ContextNamed:* and #[create|shutDown]*NamingServiceAt:)
- Facilities for dynamic object space discovery and the automatic propagation and identification of object spaces.
- Easier access to naming services via Java JNDI-like naming URLs for COS, SST and RMI naming services.
- New development additions have been added to the SST menu.
- Updates to the startUp and shutDown sequence for application contexts, spaces and invocation handlers to clean up infrastructure in a wider range of anomalous situations.
- Improved error logging and exception handling/generation.
- The ServerSupport classes have been reworked to more closely conform to the Java Servlet object model and API. Servers now support event logging.
- New classes to support frequently required server functionality (see SstDate, SstCron) and by popular demand, several previously internal classes have been made into SST API (see SstPool, SstSharedQueue, SstWorkerManager, SstRegistry, and SstDualRegistry).
- Updated Java class definitions in support of RMI interoperation and corrections to the representation of Java interfaces.
- Support for multidimensional arrays with RMI
- Java type builder changed to build Java package directory structures by default.
- Most API calls generalized to allow SstUrls, String representations of URLs or instances of SstLocalEndpoint rather than only accepting SstUrls.
- The generic SstUrl parser (not used by rmi: or http:) now substitutes the scheme as the transport should the transport be unspecified. For example, 'iiop://foo:com' is now valid.
- The TCP SstConnectCallbacks and SstDisconnectCallback callbacks now give the actual connection object (IdSstConnection) instead of the remote endpoint. This was deemed important enough to warrant breaking backward compatibility.
- New protocol to send e-mail
- SMTP now supports sending to MX forwarders through an SMTP smart host. There have been some changes to SMTP addresses too.
- The sstidlic.swp IDL cache file has been updated. Users should regenerate any of their own IDL caches. This process has been greatly simplified.
- Better error reporting on system exceptions in IIOP.
- Improved client-side handling of disconnections by an IIOP server when requests have not been serviced
- Added support for forging of IORs for custom integration with standard services in 3rd party CORBA Orbs.
- New helper methods to access named CORBA constants and exceptions from an application context.
- HTTP file servlets support modification dates.
- HTTP servlets have a default HEAD implementation. Servlets using deferred content blocks may require changes as the blocks are not fired.

- There have been changes to address generation in the Communications layer. The socket and TCP transport frameworks have had a new address-related method added too.
- The SstCommunicationMessage hierarchy's creation methods contents: and #header:contents: have been made more consistent.
- Updated and clarified categorization and specification of various API methods
- Date class>>#sstMinutesFromGMT has been obsoleted and renamed to #sstMinutesFromUTC. It will be removed in the next release.

New API and details

Helper API

A major component of the new API are methods which simplify the creation and naming of SST infrastructure for frequent use-cases. These are particularly aimed at making the application context establishment more straight-forward, eliminating the need for most configuration options and automating space creation for remote spaces. There are also much improved facilities for establishing naming services in SST, running in their own application contexts. See the sections below on the IIOP, RMI and By-Reference/Logical-Process Method Invocation helpers.

SstDate: Representing Time

SstDate represents a moment in time, a combination of Smalltalk's Date and Time. SstDate supports a subset of Date and Time creation protocol, and most of their normal protocol.

SstDate has two possible representations. A relative SstDate represents the time as a number of seconds since an epoch, an arbitrarily chosen fixed moment in time. As a result, only the local representation of the time changes with the local timezone, while the actual moment in time remains the same. For example, while 19:00 in Sydney may be 05:00 in Ottawa, it is only the local representation of the moment that is different: the actual moment in time is the same.

An absolute date, on the other hand, represents time in local time. Thus 19:00 remains 19:00 even if the local timezone shifts to daylight savings. This is the traditional representation of Date and Time in Smalltalk.

SstCron: Cron for SST

A new addition to SST is SstCron, which provides the same kind of scheduling help as Unix cron(1). With it you can schedule events to be run on the first of the month, or every Sunday, or every 5 seconds, or every 5 seconds when the first of the month is a Sunday. You can also schedule events to be run at a fixed moment such as 3:05pm on Jan 1, 2004.

To set it up you add cron entries (SstCronEntry) to cron (SstCron default) with #add: which answers the entry. To stop the entry use #remove: giving the entry. The entry has a block which is the unit of work.

The event is requeued after it has finished executed. So if you register an event to execute every second, but the event takes 5 seconds to run, it will only run every five seconds -- you won't have an event running multiple times.

The entry encodes the time constraints, with separate constraints for each of months, days, daysOfWeek, hours, minutes, and seconds. These constraints can have three possible types of values:

- nil: the constraint is true for any possible value (the default)
- collection: true for any values in the collection
- an integer: shortcut for all multiples of the integer

It's easiest to show how to use SstCron with some examples:

```
"Primitive clock on the Transcript's title bar, updating every 5 seconds.
   Inspecting the entry will show %seconds was expanded to
   (0 5 10 15 20 25 30 35 40 45 50 55)

   In Smalltalk terms this is '(0 to: 59 by: 5)'.
   So specifying a single number really means: '(minValue to: maxValue by: number)."

   SstCron default add:
     (SstCronEntry new
        workBlock: [Transcript shell title: SstDate now printString];
        seconds: 5;
        yourself)


"Run a backup every Sunday at midnight. Note that midnight
     (00:00:00) is regarded to be morning. Also note that it's
     important to specify the hours, minutes and seconds: otherwise
     they default to nil which means match every possible value.
     So without them explicitly specified you'd be running this
     event all day! We don't set the month or day cause we want
     them to always match."
    SstCron default add:
        (SstCronEntry new
            workBlock: ["perform backup"];
            daysOfWeek: #(1);
            hours: #(0);
            minutes: #(0);
            seconds: #(0);
            yourself)


"Run this on the first Monday of the month. There can only be
     one day from the 1st to the 7th that's a Sunday."
    SstCron default add:
        (SstCronEntry new
            workBlock: [foo];
            days: (1 to: 7);
            daysOfWeek: #(2);
            hours: #(0);
            minutes: #(0);
            seconds: #(0);
            yourself)

    "These should be pretty obvious"
    SstCron default add:
        (SstCronEntry
            at: (SstDate
                    newYear: 2000
                    monthIndex: 1
                    day: 1
                    hours: 0
                    minutes: 0
                    seconds: 0
                    milliseconds: 0)
            do: [Transcript shell title: 'Happy New Year!'])
```

Server support (including servlets and HTTP servlets)

Improved Java servlet conformance

Some misconceptions in the servlet implementation have been fixed. Each servlet now has its own servlet config (javax.servlet.ServletConfig, represented by SstServletConfig), while all servlets on a server share a common servlet context (javax.servlet.ServletContext, represented by SstServletContext). The previous release had only the shared servlet context (confusingly obtained through #getServletConfig) which implemented both javax.servlet.ServletContext and javax.servlet.ServletConfig protocols. This has been rectified: the servlet's config will be returned by #getServletConfig and the shared context by #getServletContext, and the protocols have been moved onto the respective classes.

ServletConfigs allow loading servlets with a custom configuration based on init parameters, a dictionary of string-to-string mappings. Relevant API are #loadServlet:as:initParameters: or #defaultServlet:initParameters:.

For backwards compatibility, SstServletConfig still supports the now-obsolete SstServletContext protocol; this will be removed in the next release.

Server logging

Servers (SstBasicServer) support a logging mechanism. The log is set with SstBasicServer>>#logger:, and the protocol depends on the server. SstBasicServer and SstHttpServer assume the logger adhered to protocol.

Servlets can use the #log: API to log messages to their associated server's log.

IIOP

Application context creation helpers

Application contexts for use with IIOP can be established with one of the new helper methods on SstApplicationContext class. Note they must still be sent #startUp before they can be used.

```
createIiopContextNamed: contextName at: location idlRepository: repository cache: cache

    "Answer an instance of the receiver for IIOP-based
    communications, named @contextName.  The context is located
    at @location, where the user wishes to be specific about
    the host:port on which messages will be serviced. The 3rd
    party IDL repository IOR is specified in @repository (as
    a filename, an IOR string or an http url) and the
    initial repository cache resides in @cache."


createIiopContextNamed: contextName cache: cache

    "Answer an instance of the receiver for IIOP-based
    communications, named @contextName.  The initial repository
    cache resides in @cache.  The answer is established at the
    default url 'iiop://' which will allocate an available port
    automatically.  All IDL information must be present in
    @cache as no remote repository is specified to dynamically
    obtain IDL definitions."
```

```
createIiopContextNamed: contextName idlRepository: repository cache: cache

    "Answer an instance of the receiver for IIOP-based
    communications, named @contextName.  The 3rd party IDL
    repository IOR is specified in @repository (as a filename,
    an IOR string or an http url) and the initial repository
    cache resides in @cache. The answer is established
    at the default url 'iiop://' which will allocate an
    available port automatically."
```

Disconnect handling

IIOP client-side disconnect handling has been tidied up for cases with uncompleted client requests. To make use of this new support when one of old-style application context creation mechanisms is being used, the method #iiopHandleDisconnects should be sent to the application context on setup. For example, in the SstPingPongIiop example, the code fragment for configuring the context is as follows:

```
context
    addSpace: #ping at: (Array with: (self urlFor: ping));
    idlRepository: ior cache: 'sstidlic.swp';    "$NON-NLS$"
    iiopInvocationScheme: self invocationScheme;
    setupFor: #ping using: SstSpaceConfiguration giopProxyConfiguration;
    iiopHandleDisconnects;
    startUp
```

```
(The new application context creation
methods #createIiopContextNamed:* do this automatically.)
```

Building custom IDL repository caches

Construction of IDL repository caches (such as the ’sstidlic.swp’ file that is distributed with the VAST 4.5 release) has been made simpler with a new API method on SstIdlRepository class.

```
buildOn: original into: destination with: elements from: repository

    "Create a new instance of the receiver, based on @original,
    with new containers whose repository ids in @repository are
    given in @elements.  The resulting cache is dumped out to
    @destination."
```

For example, creating a cache pre-loaded with the ping-pong IDL which has been loaded into a 3rd party interface repository whose IOR is in the file ’ifr.ior’, you would do the following:

```
SstIdlRepository
    buildOn: 'sstidlic.swp'
    into: 'sstpingpong.swp'
    with: #(#'IDL:PingPong:1.0')
    from: 'ifr.ior'
```

Note that to use this cache, you must make sure your IIOP application context is established to use it with the SstApplicationContext>>idlRepository:cache: method. If you have existing examples which use an alternative cache name, you can change the name of the destination cache to match your examples.

Finding CORBA values

Three new methods have been added to SstApplicationContext to make it easier to obtain local copies of CORBA objects. Naturally they will only work if you are using an IIOP application context with a repository cache which is valid. The methods are:

```
constantNamed: value

    "Answer the constant value associated with the fully qualified
    name (@value) of an IDL constant definition."

corbaConstants

    "Answer the constants from the interface repository cache,
    where these objects are associated with keys which are their
    absolute names within the interface repository. The objects
    responding to CORBAExceptionEvent protocol are also present
    in the answer."

exceptionNamed: value

    "Answer the object which responds to CORBAExceptionEvent
    protocol corresponding to the fully qualified name (@value)
    of an IDL exception."
```

Forging IORs

In certain unusual circumstances, when interacting with an ORB from a particular vendor running on a particular machine, you may wish to forge an IOR directly rather than using a naming service or a common file system. Note that this is not a safe or approved mechanism for general CORBA interactions, and relies on vendor-specific details of how they construct their object keys in IORs. If you really require this facility, the following API on SstIor class will be of assistance:

```
at: location key: key typeId: typeId

    "Answer an instance of the receiver, with its %typeId
    information set to @typeId. The profile information for the
    answer will be constructed from the @location information
    identifying the machine location and the @key information
    specifying the object's unique id at that location."
```

By reference and logical process method invocation

Application context creation helpers

Building an application context is now easier with SstApplicationContext>>#createContextNamed:at:using:. Note that the created context must still be sent #startUp before being used. createContextNamed: contextName at: location using: spaceConfiguration ″Create and setup a new instance of the receiver at @location. The new context will be called @contextName and will contain one space, a local space whose name is the #printString of @location (or @location itself if it is a String). This space will have one handler whose endpoint is at @location. @spaceConfiguration is used to configure the space created in this new context. The context created will have already been sent #setupFor:using:. Users need only #startUp the returned context.″ createSstNamingServiceAt: location ″Create, start and answer a context which runs an SST naming service at the endpoint specified by @location. It is assumed that @location is local to the receiver. If a naming service context related to @location already exists, that context is returned. Either way it is ensured that a naming service is installed.″ shutDownSstNamingServiceAt: location ″ShutDown the SST naming service (if any) running at @location.″

Disallowing global references

SstGlobalHandles, which allow addressing global objects in remote images, can now be disabled with SstSpaceConfiguration>>#allowGlobalReferences:. To preserve backwards compatibility, this currently defaults to true. Future versions will have resolution disabled by default to improve security.

Automatic space naming and discovery

Previously users had to specify names for their object spaces when creating Application Contexts. These names have to be unique within a given set of communicating components as they are used to identify locations for objects. Further, since users could specify arbitrary space names, it was difficult for the SST infrastructure to automatically discover and manage spaces.

User-defined space names are useful however where users need or want direct control over the topology of their systems. For example, if the system designer specifies four particular object spaces and no extra infrastructure, only those four object spaces will be able to interact.

In practice however, most users do not need this level of control and prefer to have the details taken care of for them. This combined with the fact that most application contexts only ever have one accessible endpoint allows SST to automatically derive space names based on the related endpoint URL. Since this URL is used to specify communications addresses, it is necessarily unique and thus the derived space names are unique.

Programmers implicitly specify that they would like their spaces automatically named simply by not giving them names when setting them up. That is, the convenience APIs (e.g., create*ContextNamed:*) do not allow users to specify space names. Using these APIs will result in spaces with auto-generated names.

Automatic space discovery is a related feature of SST. The idea is that users may not know all the participants in a distributed system when the system is setup. Rather, they would like the infrastructure to automatically manage the knowledge of various component (i.e., object spaces) which are available within the distributed system.

In previous versions of SST users needed to hook the so-called FirstContact callbacks and manually attach space specifications to their requests the first time they contacted a new object space. While this mechanism is useful, it is not adequate for automatic space discovery. It is possible to construct scenarios where logic first contacts can go undetected.

To address this problem, SST automatically includes the relevant space information when passing remote references between object spaces. There is a small amount of overhead in doing this but the benefit is that users no longer have to worry about whether or not one space knows about another space. This makes coding dynamic systems much simpler.

RMI

Application context creation helpers

Building an RMI application context is now easier. The new API below automatically creates and names the SST components according to the information provided. Note that the created contexts must still be sent #rmiStartUpWith:before being used.

```
createRmiContextNamed: contextName

    "Create and setup an instance of the receiver for
    RMI-based communications at 'rmi://'.  The new context
    will be called @contextName and will contain one space,
    a local space whose name is the #printString of the
    endpoint started at 'rmi://'.  This space will have one
    handler whose endpoint is at 'rmi://'."

createRmiContextNamed: contextName at: location

    "Create and setup an instance of the receiver for
    RMI-based communications at @location.  The new context
    will be called @contextName and will contain one space, a
    local space whose name is the #printString of @location
    (or @location itself if it is a String).  This space will
    have one handler whose endpoint is at @location."

shutDownRmiNamingServiceAt: location

    "ShutDown the RMI naming service (if any) running at
    @location."
```

Naming

Addressing objects with naming services

The addressing of objects in SST-based naming services is done using naming URLs similar to those found in Java RMI and Java JNDI. The general form of this is

```
    namingService://machine/nameToLookup
```

```
where namingService is replaced with one of rmi, sstName or cosName.
```

Given a naming url, users should ask the relevant application context for its naming interface object. For example,

```
rmiNaming
naming
cosNaming
```

gives the naming interface object for RMI, SST and COS naming services, respectively. Once you have the naming interface object, you can directly lookup naming URLs. The naming interface will automatically figure out protocols and operations to invoke to find the object for which you are looking.

A full example of using this feature is given below.

```
someAppContext naming resolve: 'sstName://foo.com/CoolServer'
```

See the comments in the API methods on SstRmiNaming, SstNamingInterface and SstCosNamingInterface for full details of the operations available and their required arguments.

Mail

API for sending e-mail

The new protocol String>>#sstEmailTo: allows easy sending of e-mail. The argument should be a URL which will specify the mail-delivering transport. The receiver should be a mail message adhering to the format specified by the mail-delivery service in the URL argument. The special 'mailto:user@host' URL format has been added as an SMTP short-cut, and mail messages (the receiver) should be in the Internet mail standard defined in RFC822. The message will only be sent to the receiver identified in the argument, regardless of any To:, Cc: or Bcc: header fields in the message.

Differences in SMTP addresses

SMTP addresses behave slightly differently now with the new address generation protocol. Addresses for remote endpoints must have a user name and host. Local addresses require neither. If a local address does not have a user name then messages are sent without a reverse-path (for errors to be returned).

SMTP smart host

The SMTP/IMAP4 transports now support sending mail through MX mail forwarders by redirecting messages through an SMTP smart host. This is configured with SstSMTPCommunications class>>#smtpSmartHost: and is a system-wide setting.

Tools

Additions to the SST menu

Three new entries have been added to the SST menu:

Trap Exceptions
Allow enabling/disabling the %trapExceptions flag on invocation configurations.

Forward Exceptions
Allow enabling/disabling the %forwardExceptions flag on invocation configurations.

Trap Marshaling Exceptions
Allow enabling/disabling the %trapExceptions flag on marshaling configurations.

The settings are propagated to all invocation handlers using these configurations as well.

Communications

Changes to address generation interfaces

#sstGenerateAddress:configuration: has been obsoleted in favor of #sstGenerateAddress:configuration:locally:. This has been made part of the IdSstAddressBuilder interface. The @locally flag allows more stringent checking to be applied by developers.

Support for #sstGenerateAddress:configuration: will disappear next release. SciSocketAddress>>#sstGenerateAddress:configuration: has been kept in the image, but all other implementations have been removed.

Changes for TCP-related addresses

Developers using custom address classes with the sockets transport framework (which includes the TCP framework) should ensure they implement the IdSstTcpRelatedAddress interface. This has a new method #sstAsUrlComponentWithHost:port:.

ObjectExtender improvements

The following improvements have been made to the ObjectExtender feature.

Integrity of meta models

Under some circumstances it was possible for the meta model data that is used internally to store the information for a Model to become corrupted. Such problems should no longer occur. Extensive error checking has been added to ensure model integrity

ObjectExtender browsers

The performance of opening the map browser has been improved.

The meta models are saved into classes that are stored in the Envy repository. It is possible to load an open edition of a class if the user switches to become the developer of that edition. If the user switches to another user it was possible to still work on one of the meta data storage entities (e.g. a model, schema, or map) and then have the changes rejected by Envy because the user was not the developer of the edition. Unless the user saw the message being written to the Transcript such an error could be overlooked and the user would be under the impression their work was saved when in fact it wasn’t. This problem should no longer occur as the browsers will not allow any of the meta model entities to be modified if the storage entity is an edition and its developer is not the current image user. It is possible to generate services from a model where the model is an edition whose developer is not the image user.

Pre-emptive progress dialogs have been added to the different generate menu options, e.g. generate on the model browser and generate services on the map browser. With the model browser generate options the user is given a list of all classes and associations in the model and has the ability to selectively choose which ones they wish to re-generate the domain classes for. This is useful if the developer has added one attribute to one class and only wishes to re-generate the domain model for this class without incurring the overhead of doing a complete model re-generation.

The inspect transactions option on the Status browser shows its transactions in a nested format. This should help to determine which child is a transaction of which parent. The printString for a transaction that is shown in the debugger stack and inspectors formerly displayed the transactions nesting level at the start in curly brackets, e.g. ″{0} SharedTransaction(Shared)″ for the shared transaction, ″{1} TopLevelTransaction(Unknown)″ or {3} Transaction(Unknown)″. The number helps determine the nesting depth of the transaction. Transactions that are committed, rolledback or marked for rollback are prefixed with C, R or MR instead of the nesting level. In addition the current transaction has its nesting level surrounded by angular brackets rather than curly ones, e.g. ″<1> TopLevelTransaction(Unknown)″. This helps to determine which transaction is current and which is not when working with a stack trace.

Mapping

There were circumstances where the mapping validation rules would report errors that two integers could not be mapped together because they did not have the same length, but the browsers did not allow integers to have their lengths changed. This has been changed such that the mapping rules allow fields that have field lengths to be mapped together as well as fields of different data types. The latter occurred when someone generated a schema as DB2 and then later changed the generation type to another format.

Code generation services

Local image generation

There were a number of problems with supporting relationships in a local image schema where methods would be generated that raised ″To be implemented″ errors. These have all been corrected such that local image services should support all the different varieties of relationships.

In addition, after applying the fixpack, it will be necessary for you to regenerate your Local Image data if you have previously generated a Local Image schema.

Model to schema generation

When schemas were regenerated, the existing ones were first discarded and subsequently replaced with new ones. Any changes to the schema such as modifying field lengths or types would have to be re-entered. Now the model will generate into any previously generated schema that has the same name as the one it would have created and for existing columns it will not modify information such as field length, type, and so on. If attributes or roles are deleted the previously generated columns are not dropped and if this is left to be managed by the user.

Schema to model generation

The role names created for each foreign key were the source and target class with a hypen between them. This meant that they did not reflect the foreign key name they were being generated from and had to often be deleted and re-created with more meaningful names. There was also a problem where two or more foreign keys between the same tables created duplicate roles with the same name. The role names now contain the foreign key logical name as well as the source and target class. In addition the roles were not being uniqued such that if an there was an existing role with the same name, e.g. as would be the case if there was more than one foreign key between the same table, the role names would be the same. They are now uniqued by having a suffix number added if an existing name is found.

Exporting schemas

The column name used for constraints was the logical and not the physical name. This has been changed to use the physical name.

Helper class for defaulting physical name lengths for schemas

A helper class, VapDatabasePhysicalRules, is provided to apply naming length conventions. The class has a singleton class instance variable which holds a maximum length. If you are using DB2 for example, you should set the maximum length to 18, as follows:

```
VapDatabasePhysicalRules maximumLength: 18.
```

What this does is apply itself to a schema, upon schema generation, such that the physical names on tables, columns, and foreign keys will be set accordingly: if the name has already been set, it will trim it down to the maximum size if necessary; it will ensure that the name is unique if it has already been used. For foreign keys, it makes a reasonable name choice, for example, the foreign key relationship name, ″Department to Customer″ would become ″Departme-Customer″.

If you have further customization requirements for a given database, you should subclass off VapDatabasePhysicalRules and customize accordingly.

Generating schemas with self-referencing associations

When generating a schema from a self referencing association, e.g. Class has Subclasses, the parent member switch in the map of the association end Class>>superclass is set to parent and not member. To correct this go into the property mapping for the associations and re-select the foreign key in the foreign key combo box. At this point you will be prompted as the end of the foreign key you are mapping to. An example of the correct way for this to be mapped is supplied in the BomSTClass in the Bom model which can be loaded by loading the Visual Age Persistence Samples maps.

Single table inheritance correction

For a model that has inheritance, when a new schema is generated a single table inheritance approach is used. However, the table maps for the primary table is created once but added twice to each class map. To correct this select one of the table maps and select the option in the browser to delete it.

Business Transaction part

For VisualAge user interfaces that have to remain open once the transaction is committed, connections had to be made to the SharedTransaction part’s beginChild to the self of a TopLevelTransaction variable. This connection has to fire when the UI opens as well as when the transaction is committed, leading to a rather cluttered free form surface. A new part, BusinessTransaction, is provided to address this problem.

In some view scenarios when a transaction is committed the developer wishes the view to remain open, in others it will be shut down. For example, an OK button might commit a view's changes and close the view whereas an Apply button might commit the changes and keep the view open.

Whenever a transaction is committed its state changes from active to committed and its parent transaction is resumed. Any attempt to resume the committed (or rolledback) transaction will cause a VapTransactionFailure exception to be thrown. The developer must therefore begin a new transaction to allow the user to remain in the view at the same transaction nesting level.

The business transaction part caters to such a scenario. It has a read only property of transaction which it will lazily initialize to a new TopLevelTransaction (Lazy initialization here means that the transaction is created when it is first asked for if it has not been previously created and is nil). When its transaction is committed (either through the commit and rollback methods on the BusinessTransaction part or by the transaction being committed elsewhere) a fresh transaction will be created by the part. When the transaction is regenerated the part signals the transaction attribute (which is bound to the transaction event). If transaction attribute of the BusinessTransaction is connected to the transaction attribute of a TransactedVariable this will cause the TransactedVariable to refresh itself with the version of its object in the new transaction and the user is allowed to continue working with the transacted variable's contents.

It is possible to explicitly set the parent transaction that the BusinessTransaction part should use to create its transaction as a child of. This is done through the writeable attribute parentTransaction on the part. This allows views that have to support nested transactions to be done by connecting the transaction attribute of one BusinessTransaction to the parentTransaction attribute of another. The latter transaction will be a nested child of the former, and the BusinessTransaction parts will ensure that whatever combination of transactions get committed or rolledback both parts have valid transactions that are nested from one another.

Informational text for model, schema, and map storage classes

The information text provided in the text view (bottom pane) of the model, map, and schema browsers has been enhanced to include more granular details. For example, the information text for a model includes not only its statistics, i.e., the number of classes, the number of relationships, and the status of the model storage class, i.e., whether is has been saved, or modified, but the information text further includes the details for each class and association in the model, that is, their attributes, oids, requiredness constraints and so on.

In addition, the information text that is displayed in the text view of the model, schema, and map browsers, is now saved in the comment field of the applicable storage class. For example, the comment field of the AutoWorldModel, one of the sample models that ships with ObjectExtender, shows the same information that is displayed in the model browser when you have the AutoWorld model selected. One of the advantages of saving the information text in the comment for the class, is that now you can compare and contrast differences between editions using the standard ″Browse changes″ tool to easily determine the delta between a given pair of models, schemas, or maps.

UML Designer improvements

A number of important fixes have been applied to the UML Designer feature: the move and copy function has been improved; some diagramming problems have been resolved, particularly with respect to using default fonts; and memory management problems have been corrected. The packaging problem which prevented the packager from functioning properly when this feature was loaded has been resolved as well.

If you are using this feature, it is strongly recommended that you install the fixpack.

Note: If you load(or reload) the UML examples, load them without their required maps. This applies only if you have installed the fixpack. If the UML examples were loaded prior to the fixpack installation, there is no requirement to reload them after installing the fixpack.

ScaledDecimal performance improvements

Significant performance improvements have been made to the ScaledDecimal>>#rounded and ScaledDecimal>>#truncate methods. These changes result in up to 18x reduction in the time taken to execute these methods. Because these methods are used extensively by other ScaledDecimal methods, a noticable overall performance improvement should be evident. The actual improvement you see in your applications will, of course, depend on the mix of methods you are using.

Chapter 3. Component fixes

This section documents fixes to problems reported to IBM by customers(APARS), and problems reported through other sources. The problems are listed according to the component to which the fix applies.

APARs

The APARs are listed according to the component to which they apply.

Base

APAR PQ12249 (9412) ″Windows″ key fires Nul for a toggle button (Windows platform)
This problem occurs only on Windows when using a Windows keyboard. The scenario is as follows: a shell containing a form, a push button and a toggle button are created. When the shell is realized the cwHandlers for push button and toggle button are Nul (value of 0). Adding a mnemonic for the pushbutton results in a KeyPressMaskEvent handler getting added to both the push button and toggle button. When the Windows key (the one with the Windows logo on it) is pressed, it causes the KeyPressMaskEvent handler to get invoked. This method calls another which processes mnemonics and accelerators. When the keysym value for the Windows key (value = 0) is compared to that of the zero-initialized toggle button, the valueChangedCallback logic for the toggle button is invoked inadvertently. A check was added to correct the problem.

APAR PQ12494 (9421) BlockContextTemplate problem in runtime images
If an instance of BlockContextTemplate is defined and returned from a method with an argument and that argument is mentioned within the block returned from that same method, and subsequently that block is placed in an instance variable of some instance, when that block-referencing instance is packaged the instance variable that referred to the block will be nil. The packager has been corrected to package instances of MethodConxtext and blocks which use MethodContext.

APAR PQ12622 (9428) Hover text not being displayed properly for some win95 controls
The tip text information in the Windows 95 controls ToolButton and TabStrip may not display some extended ASCII characters such as umlauts correctly on Windows NT 4.0.

APAR PQ15532 (10116) Textfield DBCS (Korean/NT 4.0 platform) maxlength and modification
In the following scenario: 2 text fields with maxLength: 2 and autoTab: true the behavior was encountered. Attempting to enter 2 DBCS characters. After entering 1 DBCS, you are autotabbed to the next field. If you set autoTab: false, then you can enter the two DBCS characters. For lengths greater than 2, this behavior does not occur. CwText>>#maxLength: has been corrected.

APAR PQ16281 (9622) Multilist box arrow keys (Windows platform)
In Multilist box , left and right arrow key does not slide from left to right.

APAR PQ17676 (10529) Deleting last character in System Transcript (Windows NT 4.0 Japanese)
Under Windows NT 4.0 Japanese with VisualAge V4.02B if you position the

cursor to the last character in the System Transcript and press the DELETE key, then you will get a walkback. (Index out of range).

APAR PQ18008 (10544) Fix for losingFocus event in EwTableList
Within the EwLists, when an edit is ended or cancelled, focus is unconditionally set to the list widget. This should only be done if the editWidget had focus before the end/cancel edit. A fix is provided for triggering the losingFocus event in EwTableList

APAR PQ18491 (10581) Icon label type on button disappears
When mouse two mouse buttons were pressed, a button that was displaying a bitmap or icon did not draw correctly. This has been corrected.

CIA

APAR PQ17425 (10514) Parsing large Float values in C++ causes problem. A fix is provided in the AbtCLanguageBase application.

Database

APAR PQ16493 (10456) Negative decimals saved as positive w/Static SQL. Using static SQL, a negative numeric value is being inserted as positive. AbtDatabasePackedDecimalField>># createSourceCode: has been modified to correct the problem.

MQI

APAR PQ17802 (10534) Apps packaged to MVS which use MQ. Some applications packaged to MVS that use MQ have encountered problems when using the verbs MQBACK or MQCMIT. We have fixed the problem by changing the definitions of these entry points to point to the alternative calls CSQBBAK and CSQBCMT

Web Connection

APAR PQ18687 (10591) NSAPI plugins don’t work with Netscape Enterprise Server 3.51
The NSAPI plug-ins shipped with VA 4.5 were built using Netscape Enterprise Server 3.0. The NSAPI plug-in (abtwsan.dll) was linked with ns-httpd30.lib. As a result, function calls were made to ns-httpd30.dll.

Here are some potential problems customers could experience after installing the NSAPI plug-in:

- If this was a new install of the server, an error message indicating that ns-http30.dll could not be found when trying to start the server.
- If version 3.0 of the server was previously installed, it is possible that an exception in httpd.exe would occur when trying to retrieve a VisualAge Web page.
- To correct the problem, abtwsan.dll was rebuilt using ns-httpd35.lib. The function calls are now made using ns-httpd35.dll.

Problems found by other sources

The following fixes are provided for problems found by IBM. The problems are listed according to the component to which the fix applies.

Application Builder

10344 VG/849 Environment variables larger then 512 bytes cause walkbacks (Windows NT)

EsString class>>#abtScanEnv will produce a walkback if sent to the name of an environment variable whose value is longer than 512 bytes on Windows NT. EsString>>#abtPlatformScanEnv was updated to account for environment variables larger than 511 bytes.

10528 AbtMessagePrompter #promptFor: answers self

The AbtMessagePrompter>>#promptFor: method has been corrected to return the prompt result. Formerly, it was returning self.

10538 OLE Walkback gathering type information

A walkback below occurs when the 'Shell.Application' is selected for generation in the Ole Wrapper tool 'OlexInterfaceDescription does not understand type' . The OlexInterfaceDescription>>AbtAddInterfaceInformationToSpec: has been overridden to produce the desired behavior.

10662 Attribute-to-attribute connection alignment

A modification was made to correct an alignment problem in the AbtAttributeToAttributeConnection class and its associated collaborators.

Base

10302 DBCS text input (Korean Windows/NT platform)

On a Korean Windows/95 platform, entering DBCS characters, then overwriting the same characters with different ones produced unexpected results. In the scenario tested, two DBCS characters were entered, followed by an attempt to overwrite them with two different characters. Input for only one DBCS character was accepted. This problem occurred in a packaged runtime image.

10404 Unloading ICs

Unloading ICs using System>>#removeComponent: fails because two of two missing methods.

10508 EtServerFileNamePrompter

On Unix platforms, the EtServerFileNamePrompter is dismissed whenever the Open button is selected. The fix has been added to the EtTools application.

10467 Cannot package while UML Designer is loaded

A walkback occurs when trying to package an application when UML Designer is loaded in the same image. Currently there is only a workaround for this problem which is that you must unload UML Designer before attempting to package an application.

10296 Workaround for packaging an application as a reusable image component.

If you package an application as a reusable image component, you must manually move the available prerequisite ICs over to the center pane in order to output an image.

10340 ICs walkback: attempt to store into a read-only object

When running a leaf IC (runtime IC), pool variables initialized from disk caused ″attempt to store into read only object″ errors. The problem has been corrected.

10115 Apps packaged in AIX for OS/2 target platforms fail

A PlatformFunction with a functionNumber is not output correctly when the reversing the endian of the output.

10497 Swapper packaging specifications hidden

When packaging in XD an application which has Swapper as a prereq, a walkback was encountered due to ″Undefined object does not understand asReducedString″. Subsequent investigation revealed that the problem was caused by packaging rules having hidden source code.

10504 AIX Startup scripts have bad DB2 reference

The libpath information for DB2 in the abt and abtnx startup scripts is not correct. The effect is that the DB2 shared object(s) cannot be found and abtcli.w traps. The abt and abtnx scripts have been updated.

10541 Browse Package Images option for Dev image w/ XD

When XD is loaded into an image the Tools>>Browse Package Image option is taken off the Tools pulldown menu for the Development image. This should be left available to the user so packaging can be done for a VA client app as well as for a server app in a XD passive image.

10585 Change default stack size for threads on Unix

AbtThreadManager class>>#defaultStackSize has been modified to return the greater of 16KB or AcoResourceManager class>>#defaultStackSize for Unix platforms.

10569 Double click causes walkback after NT clock rolls over

If you leave a NT machine on for an extended period of time (1-2 weeks), without rebooting, you will get a walkback when you double click inside any text window. The problem has been corrected.

10583 ScaledDecimal performance problems - #rounded and #truncated

The details for this are provided in the section on improvements and updates.

10595 Upgrade EMSRV to 6.23

In addition to the description given in the improvements section for the EMSRV refresh, the following problems were fixed:

- (NT) A GPF if the EMSRV got a connection that was immediately dropped.
- (NT) A password authentication and logon problem on various NT networking configurations when running on NT servers that were part of a domain but not the Primary Domain Controller.
- A problem with remote drive checking
- EMSRV for NetWare (both NDS and Bindery) . Some resources would not be released if the NLM was unloaded after an ’emadmin copy’ operation had been performed using emadmin 6.22 or earlier.
- NetWare for (NDS) 80 bytes of small memory allocation were no getting released after a connect and disconnect
- A file directory browser problem when dealing with a mixture of NetWare volumes supporting both long and dos namespaces
- A related bug was fixed to allow EMSRV to start up in a long file name directory

- OS/2 - EMSRV 6.23 addresses a problem where EMSRV would GPF when shutdown
- NetWare (NDS) - the browser code was enhanced to include the NetWare volumes level so users can browse between repositories on different volumes
- EMADMIN on all platforms - An extra emadmin command 'TEST' was added so that you can test if a valid repository exists on a particular EMSRV server

10589 Update descriptions for IC Packaging instructions

The description text for the 3 IC radio buttons has been updated as suggested to help customers understand when to use them, and more importantly, when not to use them.

Database

10373 Error while defining SQL statement in Composition Editor

An error occurred during SQL definition upon cancellation from a Database Connection dialog. AbtEditDatabaseQueryPartApp>>refreshPackageSpecListWith: has been modified accordingly to detect when the connection status passed to it is nil.

10590 Invalid handle - #freeEnvironmenthandle: called twice on same

There is a possibility in certain situations that the AbtOdbcDatabaseManager>>freeEnvironmentHandle: call is issued twice to the same handle, resulting in a walkback (Invalid Handle). The problem has been corrected.

Installation

10382 No MPR files installed for Server runtime. The missing files have been supplied in the fixpack

APPC

9929 CPIC Conversation part: connected event fires twice on AIX. When the connect action of a CPIC Conversation part is fired, the connected event occurs twice on AIX. In the fixpack, the extra firing of the connect event has been removed.

CICS

10492 Some of the entry points in previous CICS DLLs have changed in CICS/NT v3.1

This is causing walkbacks to occur when opening the settings notebooks on CICS parts in the composition editor. We have updated the CICS code so that it now checks for the correct CICS entry point.

10490 Allow all Smalltalk CICS transactions to use a single .ini file

Currently each VisualAge Smalltalk image running as a CICS transaction on OS/2, Windows NT, or AIX needs its own imagename.ini file to be unlocked for execution. To reduce the amount of setup work caused by the need for many .ini files, an update has been created so that a single .ini file can be used as long as it is located in the same directory where the Smalltalk images are located.

The name of the ini file for each platform is:

- **OS/2**: esvio.ini
- **Windows NT**: esvio.ini
- **AIX**: esnx.ini

After installing the fixpack, specific files are updated as noted below. Be sure to re-execute the applicable command for each CICS VA Smalltalk image.

- **OS/2**: Installing the fixpack updates the file: abtvsb40.tem Re-execute abtvbld for each CICS VA Smalltalk image.
- **Windows NT** : Installing the fixpack updates the following files: abtvsb40.tem and abtwsb40.tem Re-execute abtvbld or abtwbld for each CICS VA Smalltalk image.
- **AIX**: Installing the fixpack updates the file: abtwsb40.tem Re-execute abtwbld for each CICS VA Smalltalk image

NOTE that this fix also corrects the problem: ″Can’t run the same transaction multiple times under CICS/AIX″

MQI

10462 ABTMQS40A.W not valid for MQSeries V5

The ″willie file″ for MQSeries for AIX, abtmqs40a.w, is linked to work with MQSeries version 2. In version 5, MQSeries changed some of their lib filenames (i.e. libmqm_r4.a to libmqm_r.a). Therefore, our old willie file will not work with MQSeries version 5 when running MQ in a threaded environment. The new willie file will work correctly with MQSeries version 2 and version 5 in all cases.

10496 Correction to default MQ message creation

Whenever the method AbtMQqueue >> createDefaultMessageWithObject:withOptions: was called with a first parameter that was either a string or an AbtMQMessage containing a string, the format field of the returned AbtMQMessage was not correctly set. In some cases, this was causing a warning with reason code 2110 when the message was fetched.

ObjectExtender

10532 Meta-model integrity problems

Under some circumstances it was possible for the metamodel data that used internally to store the information for a Model to become corrupted. Further error checking has been ensure meta-model integrity.

10555 Sorting operation on removed and modified versions

If a transaction commits with removed and modified versions sorting of the versions will sort infinitely. The fix is in NewState and VersionState in VapTransactions.

10557 Service generation for queries using fillers

If you are using CLI data types and turn on the ″fillers″ style of query generation, you will get a walkback from a subclassResponsibility regarding defaultFillerValuesTable in VapCliDatabaseType. The fix is to copy that method from VapDb2DatabaseType, but if you use fillers you also need to override isValid: for each preloaded class

10558 Local image service generation includes stubs for relations

In addition to local image service support for model attributes , stubs for model relationships are generated as well

10559 Bad query gen for root/leaf with secondary tables

If a class in a hierarchy with root/leaf mapping has a secondary table, the generated queries are wrong. The problem has been corrected.

10560 Optimistic predicates for root/leaf mapping breaks queries

If an optimistic predicate is specified for a class with a multi-table mapping, the predicate is applied to all of the update queries, making most of them invalid.

105621 Generating model<>schema does not merge with existing models or schemas

When generating a schema from model or vice versa, the generation ignores any existing model or schema and creates a new one. The problem has been corrected.

10562 User query helpers are broken

The methods manyQuery:with: and singleQuery:with: in RelationalServiceObject are incorrect. They need parentheses around (self defaultInjection: anObject), and they need to send ″aQuery raiseErrorFlagOn″ and ″self setToCompleted″ just like the other basic queries:

10564 Discriminator with physical name

If a discriminator is mapped to a column with a different physical and logical name, the logical name is used to remove the column from the insert data method on the DO, and generates an ″at: 0″. The fix is to remove the discriminator column by physical name.

10578 Mapping to self referencing foreign-keys

If you have a foreign key which is self referencing, i.e., Class.subclass, tell the map browser which end you are referring to when you map to it. The fix is to prompt the user for the key end when a self-referencing foreign key is chosen.

10587 Local Image find with non-existent key causes error

A non-existent key for a given model class causes an error in the local image persistence find path.

10600 The DDL for constraints use the logical name and not the physical

The DDL generated for foreign keys used the logical name, which could include spaces or exceed the max length. The physical Name should be used.

10601 Model to schema generation includes spaces in names of foreign keys

When generating a schema from model, the names of foreign keys include spaces. Subsequently when they are exported they will generate errors.

10602 More than one foreign key between 2 tables

On Schema to Model generation, if there are more than one foreign keys between the source and target classes the generated roles are the same. Also, the mapping of the two relationship points to the same foreign key, because the lookup by name fails to differentiate the two

Workaround for root-leaf inheritance scheme

If your model follows the root-leaf inheritance scheme. The generated insert queries for the leaf classes are executed in the wrong order. If constraints are turned on in your database, the inserts will result in an error. To fix this, just reverse the order of the insert statements.

Example:

```
insertQuery: argArray withInjector: anInjector
  "Return an array of query specs for the query called insert"

   ^(Array new:
      at: 3 <<=====Used to be 1
      put: ( AbtQuerySpec new
              statement: ((SqlQueryString string: self insertRACECARSqlString)
              evaluate: (anInjector convertAll: (argArray at: 1))));
      at: 2
      put: ( AbtQuerySpec new
              statement: ((SqlQueryString string: self insertAUTOMOBILESqlString)
              evaluate: (anInjector convertAll: (argArray at: 2))));

      at: 1 <<=====Used to be 3
      put: (AbtQuerySpec new
             statement: ((SqlQueryString string: self insertVEHICLESqlString)
             evaluate: (anInjector convertAll: (argArray at: 3))));
      yourself
```

Workaround for Weakness problem

IBM Smalltalk uses an instance of EsQueue to hold objects that are weakly collected by the garbage collector. During a garbage collect any weak object whose contents are not referenced by any other objects (unless they themselves are weak) are placed on the queue, together with the weak object and the instance variable slot number that the now GC'd object occupied. In image idle time the triples are read off the queue and the weak object sent the message finalizedSlot:removedObject: with the parms of the ivar number and the weakly referenced object (the slot is set to nil in the GC). The problem is that if the queue runs out of room during a GC then no more objects are processed. This means that the image can be in an unstable place where one reference to an object has been set to nil and another reference has not.

To overcome this the size of the queue should be set to a big number. The packager by default makes the queue very small in images that are packaged (even if the queue has grown quite large in the packaging image) so this problem only manifests itself in packaged images. Until a better fix is found it is probably a good idea to specialize a startUp method on an application to force the queue size, e.g.

```
MyApplication class>>startUp
  super startUp.
 ( Core registry at: 37 ) isEmpty ifTrue: [ ( Core registry at: 37 ) size: 999 ].
```

10630 Duplicate cluster maps are generated

Generating a schema from a model with inheritance, duplicate cluster maps are added to the subclasses. The problem has been corrected.

10631 Generate stubs create incorrect service methods

The generate service methods do not take the correct parameters and will not be called even if completed. The problem has been corrected.

10632 Walkback occurs generating stubs or service layer for local image services.

The problem has been corrected.

10633 Generate model to schema with self-referencing result in bad parent code

Generate a schema from a model with a self-referencing association, i.e. -BOMmodel, and the relationship maps are both set to #parent, will not generate queries correctly. The problem has been corrected

10634 Incorrect schema from model with inherited

If you generate a schema from a model with a composed attribute in a parent class, duplicate columns will be added to the subclass' table. The problem has been corrected.

10637 Walkback occurs after adding a key to an imported table without a key

Import a table with no primary key defined via the schema browser-> import schema. Edit the table and add a column to the empty primary key list. Generate the code and a walkback will occur. The problem has been corrected.

Code file-in fix for editing property maps in class for new data store map or new class in existing map

When editing the property maps for a class in a new data store map or for a new class in an existing data store map, you may receive the following walkback: ″VapValidationError does not understand attributeNamed:″. File in the following code to fix it:

```
!VapRelationshipMap publicMethods !

validate

   "Verify that the receiving attribute map is complete and correct
    enough for service generation.
    Test that:
      - our association exists
      - our association end exists
      - our type map exists and is valid
      - our foreign key exists
      - if we have an inverse map it is mapped to the same foreign key

    Returns true or an VapErrorObject describing the failure."

    | rc key inverse |

    (rc := self association) isVapError ifTrue: [^rc].

    (rc := self propertyEnd) isVapError ifTrue: [^rc].

    (rc := self typeMap) isVapError ifTrue: [^rc].

    (key := self foreignKey) isVapError
       ifTrue: [^key]
       ifFalse: [(rc := key validate) isVapError ifTrue: [^rc]].

   ((inverse := self inverseMap) notNil)
     ifTrue: [
       (inverse foreignKeyName ~= self foreignKeyName)
  ifTrue: [^self errorInverseMap: inverse]].
   ^super validate   ! !
```

Report Writer

10513 Field breaks should have parentage option

If Field1 break occurs before Field2 break, then a break should be forced in Field2 each time a break occurs in Field1. Currently, there is no interdependence. Breaks for the different fields occur independently of one another. CoFieldBreak and AbtFieldBreak have been modified to include a new attribute #parentTriggersBreak. The attribute is a boolean which determines whether a field break should occur if a preceding field break occurs in an iterator.

Server Runtime

10566 Improve Smalltalk CALL-IN Performance It has been reported by customers that the CALL-IN support for Smalltalk is slow. After looking at the code it was determined that performance improvements could be made by reducing the amount of data movement and by caching the look up of class objects and selectors.

Server Workbench

10375 Installed features must reference VA Smalltalk run

Now that VA non-visual parts are allowed to be installed in a XD image, a Feature should not show up on the Installed Features list until both the VisualAge and Smalltalk features are imported into the manager. Currently if only the Smalltalk feature is imported, the feature shows up in the list. If you try to install it into a passive image you get error message such as ’AbtCommunicationsRun has no editions’ and/or ’AbtCommunicationsRun V 4.5 unknown not found. Choose another one’.

10594 Server sample app file abtmvssp.dat missing from v4.5

The Server Sample file abtmvssp.dat referenced in the Server guide on page 19 is missing from V4.5 on the CD.

10604 Upgrade SST

In addition to the description given in the improvements section for the SST refresh, the following problems were fixed:

- GIOP marshaling of integers when passed to float/double types doesn’t do autoconversion
- Worker processes still alive after a SstWorkerManager shutDown
- OBJ - #spaceHosting: does not look at local endpoints
- Walkback finalizing invalid endpoints
- SstSimpleObjectSpace>>#export: doesn’t answer export-key
- RMI - Handle multi-dimensional arrays
- SstInspector’s ’Browse class’ can cause remote message sends
- SstSocketTransport leaks sockets on connect failure
- RMI - LookupTables do not marshal correctly
- RMI - #sstMarshalingValue is sent to objects being loaded
- SstInvocationHandler silently fails on #receive error
- SstSimpleObjectSpace>>#export: doesn’t answer export key
- SstPackagingInstructions unnecessarily specifies EpXDPackaging as a prereq
- RMI - Java interfaces cannot extend multiple interfaces
- Remote administration only auto-updates once
- OBJ - add*Space* methods need reanalysis for thread safety
- SstNamingService and SstCosNaming - compound name resolution problems
- SstCosNamingContext>>parent: not understood by remote naming contexts
- SstCosNamingContext>>bindCompound:obj: fails to bind when part naming context object is remote
- SstCosNaming - failures whenever remote CosNaming objects mixed with local SstCosNaming objects

- RMI - schemes other than ’rmi:’ are not handled correctly
- COMM - #asSpaceName and #asLocator missing from URL
- SstCosNamingContext>>list:bl:bi: fails with remote contexts
- #clear can cause walkbacks due to a context’s properties being nil
- KERN - Spurious ACO failures
- SstAbstractRequest sends initialize twice on #new
- RMI - Walkback: UndefinedObject does not understand #requestReceiver
- IIOP - #locateRequestFrom:using: does at:ifAbsent: with nil
- RMI - random hangs with #invoke:at:
- DOC - misnamed category in IuSstMarshalable
- IIOP - Indirection type code unmarshaling can fail
- IIOP - struct/union/enum/exception defs not present in interfaceDefs

UML Designer

The following problems have been corrected

- (9761) Edit/undo function walkback
- (10267) Translation errors on Brazilian and Japanese platforms
- (10273) Published index page correction for Japanese platform
- (10349) Errors when unloading UML Designer

Note: If you load(or reload) the UML examples, load them without their required maps. This applies only if you have installed the fixpack. If the UML examples were loaded prior to the fixpack installation, there is no requirement to reload them after installing the fixpack.

Web Connection

10223 #BUTTONTYPE attribute should not be translated
The #buttonType attribute of an AbtHtmlPushButton is currently string separated in the generated runtime code (#abtBuildInternals). This attribute should not be translated since its value is included in the rendered HTML. A #isAttributeTranslatable: class method has been added to AbtHtmlPushButton to correct the problem

10520 Connection lines don’t update on table column resize
Connection lines are not redrawn when the mouse pointer is used to resize the width of a column in an AbtHtmlTable. A fix has been added to the AbtWebEditBase application

10556 Walkback opening empty subclass of web part
Attempting to open a new subclass of web part results in a walkback stating that no such subpartBuilder ″Page″ exists. A fix has been applied to both edit and runtime applications, AbtWebEditSmalltalk, and AbtWebRunSmalltalk

10567 Resize to fit doesn’t shrink to fit
Clicking the ’resizeToFit’ button of an HtmlPage does not cause the page width to become smaller when applicable. A fix has been added to the AbtWebEditBase application.

10596 Excessive repainting using Web parts on AIX
Grid parts were updating their selection rectangle even though the rectangle

had not changed. Selection of certain web parts was causing a total refresh of the entire page. ResizetoFit logic has been updated to eliminate extra refresh calls

Chapter 4. Packaging ICs

If you package your applications using ICs, and you have installed the fixpack, the following notes apply to you. When you install the fixpack according to the instructions we have provided, certain IC files are overwritten intentionally. In addition, a new subdirectory of ICs are created. To package with the new ICs, you will need to follow certain steps. These steps are explained later. First, a few details are given regarding what occurs during the installation as it pertains to the IC files.

Notes:

1. Unpacking the fixpack on top of a client causes certain snapshots to be overwritten in the **icspkg**directory tree. The modified snapshots will be the ones corresponding to code changes in the fixpack. You will not be able to package ICs from that client image until the new fixpack code has been loaded into the image. (This is because the loaded code - in the image - must match the timestamps inside of the snapshots in the icspkg tree.)
2. Unpacking the fixpack on top of a client causes a new runtime IC directory to be created: **ics45a**. This directory contains the set of new ICs which replace their counterparts in the **icsrun** directory (FOR THE PURPOSE OF RUNTIME IMAGES ONLY). You MUST NOT COPY these ICs over top of their icsrun directory or else their development image will not be able to start up. (The development image is married FOREVER to the original set of ICs that we shipped with Version 4.5)

To utilize the new ICs in your runtime applications, you can use one of the following two techniques:

Technique 1. In the **ini** for the runtime IC image, edit the line for each of the new ICs to point to the IC in the **ics45a** directory instead of the icsrun directory.

Technique 2. Follow these steps in the order they are given.

1. In the **ini** for the runtime IC, remove all IC paths from the IC PATH section.
2. Copy all of the ICs from the **icsrun** directory to the runtime application directory.
3. Copy the ICs from **ics45a** into the runtime application directory.

   By default, ICs are found first in the image's working directory unless there are explicit path statements to them in the **ini** file.