TIBCO ActiveSpaces®

Release Notes

Software Release 2.1.2 Add-On
January 2014

Two-Second Advantage®

Important Information

Contents

Release Notes

Check the TIBCO Product Support web site at https://support.tibco.com for product information that was not available at release time. Entry to this site requires a username and password. If you do not have a username, you can request one. You must have a valid maintenance or support contract to use this site.

Topics

New Features

This section lists new features of each release of TIBCO ActiveSpaces.

2.1.2 Add-On Release

The TIBCO ActiveSpaces 2.1.2 Add-on release has the following new features.

- **New C Admin Commands** The ActiveSpaces 2.1.2 Add-on introduces a new C version of the as-admin utility—`as-admin.exe`.

  This utility is located in the `AS_HOME/bin` directory and can be invoked by navigating to the /bin directory and entering:

  ./as-admin

  The C Admin utility includes metaspace commands, space commands, admin routing commands, logging commands, and miscellaneous command. For information on the new commands, see the A*ddendum to Readme for TIBCO ActiveSpaces 2.1.2 Add-on* document.

- **TIBCO ActiveSpaces Hawk Microagent** An ActiveSpaces managed application instrumented with the TIBCO Hawk Application Management Interface (AMI) protocol. The Hawk microagent exposes a set of methods that allow the connected Metaspace to be monitored and controlled by TIBCO Hawk tools. The ActiveSpaces Hawk Microagent communicates with the TIBCO Hawk Agent by using TIBCO Rendezvous as the transport for exchanges of monitored and management data to interact with the Metaspace.

  For information on the ActiveSpaces Hawk Microagent, see the A*ddendum to Readme for TIBCO ActiveSpaces 2.1.2 Add-on* document.

- **Cross-Site Replication** The cross-site replication feature lets you deploy ActiveSpaces across multiple data centers and have the data replicated between them. Cross-site replication provides a single view of all the data, fault tolerance at the data-center level, and faster performance through the reduction of round-trip times for clients located near the data centers.

  For information on the cross-site replication feature, see the A*ddendum to Readme for TIBCO ActiveSpaces 2.1.2 Add-on* document.

Release 2.1.2

TIBCO ActiveSpaces Release 2.1.2 has one new feature:

- **ActiveSpaces Routing** Allows you to set up router nodes that send updates from one customer site using any transport mechanism; for example, TCP/IP or an Enterprise Messaging System (EMS).

  The routing interface can also be a remote client connection from one site to another or an other transport mechanism that your application implements.

  For detailed information on implementing the routing feature, see the *Addendum for TIBCO ActiveSpaces 2.1.2 Add-on* document.

Release 2.1.1

TIBCO ActiveSpaces Release 2.1.1 has the following new features:

- **ActiveSpaces Security** ActiveSpaces now provides security features that allow you to set up a secure ActiveSpaces cluster. You can:
  - Set up nodes as security domain controllers, which control security within a domain, or as security domain requestors, which request access to a metaspace controlled by a security domain controller.
  - Use the ActiveSpaces Admin CLI and API sets to configure security settings by creating a security policy file.
  - Enforce security by generating security token files and distributing them to security domain requesters that need to connect to specified metaspaces in the secure cluster.
  - Implement user authentication using operating system authentication such as NTLM/Active Directory-Kerberos or UNIX/Linux Pluggable Authentication Modules (PAM) login; or you can use LDAP authentication.

    You can also use X509v3 authentication based on verification of the user's X509 certificate against an LDAP user account using the Simple Authentication and Security Layer (SASL) framework.

— Use Access Control Lists (ACLs) to provide granular access control to metaspace/space objects and operations.

— Encrypt tuple data (except for key fields and indexes) both in memory and when persisted to local storage.

The ActiveSpaces 2.1.1 security feature is not supported with the ActiveSpaces Monitoring and Management (ASMM) interface.

For detailed information on ActiveSpaces security, see chapter 4 of the *TIBCO ActiveSpaces Developer's Guide*, "Implementing ActiveSpaces Security."

- **Affinity Feature** The Admin CLI and the ActiveSpaces API set now allow you to define tuple fields as distribution fields.

  Rather than carrying out distribution based on the values of *all* of the key fields, affinity allows you to distribute only on a subset of the key fields. All of the records with equivalent values for their distribution fields are seeded by the same node.

  The **`define | create space`** command in the Admin CLI contains a new parameter, **`distribution def`**, which allows you to define distribution fields. Distribution fields must be already defined in the key for the space.

  Two new functions have been added to each API set to set and get distribution fields.

- **Host-Aware Replication Feature** ActiveSpaces now allows you to run seeders in groups to help prevent the loss of replicated data. When groups are used, the data from the seeders in one group will always be replicated on seeders that are in other groups. If a device hosting the seeders in a group goes down, no data loss will occur as the replicated data is guaranteed not to reside on any of the seeders in that group.

  Running seeders in groups to prevent the loss of replicated data is useful when:

  — Several seeders reside on the same physical entity (e.g. a computer or rack)

  — Access to several seeders is thru the same physical entity (e.g. a network switch)

  For detailed information on the feature, see Host-Aware Replication in the *TIBCO ActiveSpaces Administration Guide*.

Deprecated Features:

- With release 2.1.1, multicast discovery using TIBCO Rendezvous as a transport is deprecated.

- With Release 2.1.1, support for the following platforms is deprecated:
  - **Red Hat Enterprise Linux AS** 4.x 32-bit on x86 and 4.x 64-bit on x86-64
  - **Red Hat Enterprise Linux ES** 4.x 32-bit on x86 and 4.x 64-bit on x86-64

Release 2.0.2

Release 2.0.2 has the following new features:

- **Add Field** ActiveSpaces allows users to add fields to a space that is already defined and in use by using the `as-admin` utility or by calling `Metaspace.alterSpace()`. There is no disruption of service. New fields must be nullable.

  If the space has not yet been defined or the space definition is incompatible with the one that is defined (e.g. has new fields that are not nullable), an exception is generating describing what was incorrect.

  The as-admin utility also adds a command line parameter to alter a space.

  For information on alter space, see Adding Fields to a Previously Defined Space in the *TIBCO ActiveSpaces Developer's Guide*, the JavaDoc for `Metaspace.alterspace()`, and the reference topic on the `tibas_MetaspaceAlterSpace()` in the *TIBCO ActiveSpaces C API Reference*.

  For information on the `as-admin` **alter space** command, see alter space in the *TIBCO ActiveSpaces Administration* document.

- **Add/Drop Index** ActiveSpaces allows you to add or drop indexes to and from a space that is already defined and in use by using the as-admin utility or by calling `Metaspace.alterSpace()`. There is no disruption of service. New indexes are built in the background and are used automatically to optimize queries when ready.

  The `as-admin` utility provides a new **alter space drop index** command that allows dropping indexes and an **alter space add index** command that allows adding indexes.

  For information on the C API function (`tibasSpaceDef_RemoveIndexDef()`), see the reference topic on `tibasSpaceDefRemoveIndexDef()` in the *TIBCO ActiveSpaces C API Reference*. For information on the Java method, see the JavaDoc for ActiveSpaces 2.0.2.

  For information on the `as-admin` commands (**alter space drop index** and **alter space add index)**, see the *TIBCO ActiveSpaces Administration* document.

- **New as-admin Statistics** The **show member** and **show space** commands in the `as-admin` utility now report statistics for locks, unlocks, queries, and invokes.

For information on the show member statistics, see show | describe member in the *TIBCO ActiveSpaces Administration* document.

- **Support for AIX 7.1** ActiveSpaces now supports AIX 7.1.
- **Support for SUSE Linux Enterprise Server** ActiveSpaces now supports SUSE Linux Enterprise Server version 10 and 11.
- **Support for Java 7** ActiveSpaces now supports Java 7 (JDK 1.7.0_7 or higher for Windows and JDK 1.7.0_04 for Linux).

Deprecated Features:

- The default index type for primary keys is no longer `TREE`. The default index types are now:
  - `HASH` for primary key index
  - `TREE` for secondary indexes

Release 2.0.1

Release 2.0.1 has the following new features:

- **Space Wait, Write and Read Timeouts** Added functions to set and get space wait timeouts, read timeouts, and write timeouts.

  Space wait timeouts are applied to operations that cannot be processed because a space in not in the READY state, i.e., the space is in the INITIAL, LOADING, RECOVER, or SUSPEND state. The timeout value specifies how long the operation waits for the space to become ready before returning an error.

  Write timeouts are set for a specified space and applied to Put, Take, Lock, and Unlock operations.

  Read timeouts are set for read operations (Get operations).
- **Context Functions** Added functions to implement thread contexts that enable one thread to release locks and transactions and another thread to acquire the locks and transactions.
- **Logging to a File** Added support for logging to a file.
- **Retrieval of Member Join Time, Host Address, and Port Number** Added the ability to retrieve the time a member joined the metaspace, its host address, and its port number through the API and display these in ASMM and in the `as-admin` tool.
- **New Logging Level** Added the log level NONE.

- **CloseAll() Method** Added closeAll() method to the metaspace interface. `closeAll()` forces closure of the metaspace and release of its resources.
- **Configurable Worker Thread Count** Added the ability to specify the number of worker threads a member can use for remote invocation.
- **Retrieval of Member Join Time, Host Address, and Port Number** Added the ability to retrieve how long ago a member joined a space, its host address, and its port.
- **Remote Invocation of Functions/Methods** Added remote invocation methods that take a string containing the name of the class that implements the `Invocable` or `MemberInvocable` interface.
- **Remote Listener** Added the ability to listen for events on remote members.
- **as-convert Utility** Added utility for upgrade of data store files for shared-nothing persistence. This utility is used to migrate any existing shared-nothing persistence data store files to the proper format required by AS 2.0.1. For detailed information, see Running the as-convert Utility on page 17 of the *TIBCO ActiveSpaces Installation* document.
- **Ability to Set Different TTL for each Entry** Added the ability to set a different TTL for each entry using the Put options.
- **Ability to Disable the ASMM Utilities Menu and Space Browser** Added an entry in the clientconfig.xml file to allow disabling of the Utilities menu and the space browser.
- **.NET API Help** Added a .NET API help system that can be viewed on Windows platforms.
- **Documentation Fixes** Fixed issues with the HTML code for the *TIBCO ActiveSpaces Installation* document and an issue with the title page for the *TIBCO ActiveSpaces API Reference*.

Deprecated Features:

- Use of the `takeTransaction` and `releaseTransaction` methods of the Metaspace object have been deprecated.

  These methods have been replaced with the `takeContext` method (**`tibasMetaspace_AcquireContext() f`**or the C API) and with the `releaseContext` method (`tibasMetaspace_ReleaseContext()` for the C API).
- Use of the `disconnect()` method on the metaspace has been deprecated. You should now use the `metaspace close()` method

- Use of a lock scope has been removed from the following: lock options, put options, take options.
- The following operation status settings have been deprecated and consolidated into the TIBAS_INVALID_ARG status setting:
  - INVALID_URL
  - INVALID_TUPLE
  - INVALID_SPACE
  - INVALID_RESULT
  - INVALID_PERSISTER,
  - INVALID_OP
  - INVALID_NAME
  - INVALID_METASPACE
  - INVALID_MEMBER
  - INVALID_LOCK
  - INVALID_LISTENER
  - INVALID_ITER
  - INVALID_FILTER
  - INVALID_EVENT_BROWSER
  - INVALID_EVENT
  - INVALID_ENTRY
  - INVALID_DEF
  - INVALID_DATETIME
  - INVALID_COLLECTION
  - INVALID_BROWSER
  - INVALID_ADMIN
  - INVALID_ACTION_RESULT
  - INVALID_ACTION
  - TIBAS_MISMATCHED_LOCK
  - NOT_LOCKED

Release 2.0.0

Release 2.0.0 has the following new features:

- **.NET support** Added a .NET API and a .NET `as-agent` for development and execution in the .NET environment.
- **Shared-nothing persistence** Includes built-in persistence based on shared-nothing architecture where data can be written to a file system that is either on a local disk, SAN, or any other network storage device, for horizontal scalability.
- **Remote client** Allows ActiveSpaces applications to exist remotely across a WAN instead of having to exist locally on the LAN. To implement remote clients, you must purchase licenses for the TIBCO ActiveSpaces Remote Client.
- **TIBCO ActiveSpaces Monitoring and Management (ASMM)** Allows users to connect to a metaspace, display information about various ActiveSpaces system resources, for example, metaspaces, metaspace members, spaces, entry statistics, and so on, and browse space entries.
- **Indexing and query optimizer** Allows the user to create secondary indexes that can be a single field or multiple fields for a composite index. Added support for both TREE and HASH indexes. Also added query optimizer that automatically selects which indexes to use for the best performance.
- **Remote invocation of functions** Allows users to invoke functions from a client and execute them on members.
- New Example Functions

Release 1.1.1

This release did not have any new features.

Release 1.1.0

The following are new features in this release.

- **Batch operations** Batch versions of the Space operations are now available. Batch operations are parallelized and therefore usually result in higher throughput than invoking multiple individual operations in a serial manner.
- **Persistence interface** It is now possible to mark a Space as persisted to a permanent persistence layer, and for users to register their implementation of the persistence interface to the persistence layer of their choice with the Space.

Example implementations of the persistence interface are provided in the examples (using JDBC in Java, and using the PostgreSQL PQ library in C).

- **Space capacity and capacity policy** It is now possible to define a maximum capacity for a Space (expressed in number of entries per seeder) and to select a behavior (the capacity policy) when the maximum capacity is reached by a seeder. The behavior can be either to evict another entry using a **Least Recently Used** selection algorithm (allowing the use of a Space as a cache) or to throw an exception (signifying that the Space's maximum capacity has been reached).
- **Minimum number of seeders** It is now possible to specify a minimum number of seeders to be joined to the Space before the Space can be used. This number must be equal to or greater than 1 (the default value being 1).
- **Space life cycle** Spaces now have a life cycle. After being defined and before the right number of seeders (and persisters) is reached, the Space is in the **initial** state. If the Space is persistent, it will then change to the **loading** state. Once the Space has reached the desired number of seeders — and, if persisted, it has at least one registered persister — and has gone through the loading state, it reaches the **ready** state and can be used. Applications can check for the Space being ready or wait until it becomes ready using the new `isReady()` and `waitForReady()` operations.
- **New way to specify the key fields for a Space** Marking fields as keys for a Space has moved from the `FieldDef` level to the `SpaceDef` level. The `FieldDef` object's `setKey()` and isKey() methods are now deprecated, and new `SpaceDef setKey()` and `SpaceDef getKey()` methods replace them.
- **Space update protocol selection** It is now possible to select for each Space between using a unicast or a multicast network protocol to propagate events generated by changes in the data stored in the Space.
- **Admin language grammar update** The arguments to the `define space/create space` command have been homogenized to all follow the `attribute=value` format.
- **Common object** A new `Common` object is introduced that allows access to all of the Metaspace connections for the process.

Java API changes

Individual (that is, non-batched) Space operations now return a `SpaceResult` object rather than a `SpaceEntry` object and do not throw checked exceptions anymore. The `SpaceResult` object may contain an `Entry` and may contain an `ASException`. It is also possible to retrieve directly the `Tuple` contained in the `Entry` (if there is one) from the `SpaceResult` object (therefore simplifying the task of porting source code from the 1.0 version of the API to the 1.1 version).

C API changes

Collections functions The collection convenience functions have changed from `Iter` to `List` (for example `tibasStringIter` functions are now replaced with `tibasStringList` functions). Additional `List` functions have been introduced as needed by batch operations: `tibasTupeList`, `tibasEntryList`, and `tibasResultList`.

Bug fixes

The limitation of 1 MB maximum Tuple size when using the PGM reliable multicast transport has been removed.

Release 1.0.2

This release has no new features.

Release 1.0.1

This release has no new features.

Release 1.0.0

TIBCO ActiveSpaces is a peer-to-peer distributed in-memory data grid. It allows developers to create distributed applications that exchange and modify data shared between processes and across a network.

ActiveSpaces combines some of the important features of a database and a messaging system in a single integrated interface.

Like a Database

- Has a data definition language
- Has SQL-like where clause filters
- Can be made fault-tolerant using data replication
- Implements a form of horizontal partitioning
- Has locks for concurrent access control
- Includes support for transactions
- Supports CRUD operations (Create, Read, Update, Delete)

Like a Messaging System

- Listeners give applications the ability to subscribe to changes in the data

- One or multiple recipients
- Changes to the data are immediately distributed to all intended recipients at the same time.
- Browsers let applications use a Space as a queue.

Administration

ActiveSpaces includes a utility called the Administration Command Line Interface (Admin CLI), which allows the administrator to connect to a Metaspace, create a Space, and display information about existing spaces and members. For further information please refer to the Administrative Commands section in *TIBCO ActiveSpaces User's Guide*.

Additional Features

Beyond the simplicity and convenience of having a single unified interface for both data storage features and messaging features, ActiveSpaces offers these additional features:

- The ability to receive *initial values* when creating a listener.
- The ability to run *continuously updated queries* in real-time.

Changes in Functionality

This section lists changes in functionality since the last major (2.1.1) release of this product.

Release 2.1.2

Release 2.1.2 adds new C functions, Java methods, and .NET API functions to support the ActiveSpaces routing feature.

C Changes

The Put options and the Take options now have an extra boolean element—`tibas_boolean route`—indicating whether a specified put or take operation should be routed.

```
struct _tibasPutOptions {
    tibas_long entryTTL;
    tibas_long lockWait;
    tibas_boolean lock;
    tibas_boolean unlock;
    tibas_boolean forget;
    tibas_boolean route;
    void* resultHandler;
    void* closure;
};
```

```
struct _tibasTakeOptions {
    tibas_long lockWait;
    tibas_boolean lock;
    tibas_boolean unlock;
    tibas_boolean forget;
    tibas_boolean route;
    void* resultHandler;
    void* closure;
};
```

Added:

- tibasSpace_SetRouter()
- tibasSpaceDef_SetRouted()
- tibasRouter_Create()
- tibasRouter_Free()

Deprecated:

- tibasSpaceDef_SetPersisted()

Java Changes

A new Router interface has been added to support the ActiveSpaces routing feature. The Router interface is defined as follows:

```
public ActionResult onOpen (RouterOpenAction openAction);
public ActionResult onClose (RouterCloseAction closeAction);
public ActionResult onWrite (RouterWriteAction writeAction);
public ActionResult onAlter (RouterAlterAction alterAction);
```

Added:

- SetRouted()
- IsRouted()
- setRouter()
- stopRouter()

.NET API Changes

Added:

- Router.Router Router { set; }
- public abstract bool Routed { get; set; }

  (SpaceDef method, similar to setRouted)

Release 2.1.1

Change in Default Nullable Setting for Space Fields

With release 2.1.1 as with the previous release (the 2.0.2 HF-001 release), the default setting for the as-admin Admin CLI command for defining fields with the **`define | create`** `space` command or altering a space with the alter space command is "not nullable."

In prior releases, the default value was `nullable`. The default setting for defining or altering fields using ActiveSpaces API calls remains "not nullable."

C Changes

The signature of the `tibasAdminExecute()` function has been changed. The result parameter is now a pointer to a character buffer that ActiveSpaces allocates to hold the result string.

The new signature is:

```
tibasAdmin_Execute(tibasAdmin admin, char** result, tibasMetaspace
metaspace, const char* cmd)
```

New functions have been added to support the ActiveSpaces security feature and the affinity feature.

Change to Default timescope setting for Browser Definitions

The default timescope value for the BrowserDef object has been changed from ALL to SNAPSHOT.

Change to Default timescope setting for Listener Definitions

The default timescope value for the ListenerDef object has been changed to `TIBAS_TIME_SCOPE_NEW_EVENTS`.

Added:

- tibasMemberDef_GetSecurityPolicyFile()
- tibasMemberDef_GetSecurityTokenFile()
- tibasMemberDef_GetAuthenticationCallback()
- tibasMemberDef_SetSecurityPolicyFile()
- tibasMemberDef_SetSecurityTokenFile()
- tibasMemberDef_SetAuthenticationCallback()
- tibasFieldDef_IsEncrypted()
- tibasFieldDef_SetEncrypted()
- tibas_SetSecurityLogLevel()
- tibas_GetSecurityLogLevel()
- tibasSpaceDef_SetDistributionFields()
- tibasSpaceDef_GetDistributionFields()
- tibasMemberDef_SetDataStore()—new for the C API only
- tibasMemberDef_GetDataStore()—new for the C API only

- tibasSpace_GetMetaspace()
- tibasSpace_GetMetaspaceName()

Changed:

- tibasAdminExecute()
- tibasInvokeResultList_GetResults() has been renamed to tibasInvokeResultList_GetReturns().

Java Changes

Added:

- String MemberDef.getSecurityTokenFile()
- String MemberDef.getSecurityPolicyFile()
- AuthCallback MemberDef.getAuthenticationCallback()
- MemberDef MemberDef.setSecurityPolicyFile(String policyFile)
- MemberDef MemberDef.setSecurityTokenFile(String tokenFile)
- MemberDef MemberDef.setAuthenticationCallback();
- FieldDef.IsEncrypted()
- FieldDef.SetEncrypted()
- setDistributionFields()
- getDistributionFields()
- ASCommon.setSecurityLogLevel()
- ASCommon.getSecurityLogLevel()

.NET API Changes

- FieldDef.Encrypted
- ASCommon.SecurityLogLevel
- SetDistributionFields
- GetDistributionFields
- MemberDef.DataStore Property
- ASSecurityException
- ASSecurityProvider class

- AuthenticationInfo class
- Abstract Credential class
- UserPwdCredential class
- X509V3Credential class
- AuthenticationCallback Interface

Admin Language Changes

The following new commands have been added to support the ActiveSpaces security features:

- **define | create security_policy** Creates a security policy file that is used to configure security settings for a security domain.
- **define | create security_token** Creates a security token file that is deployed to requestor nodes that need to access a metaspace in a security domain controlled by a security domain controller.
- **validate policy_file** Validates the syntax of a specified security policy file.
- **validate token_file** Validates a specified security token file.
- **validate truststore** Validates a specified security certificate file.

The following commands have been changed:

- The **`connect`** command has a two new parameters to support connection as a security domain controller or a security domain requestor:

  ```
  [security_token <string>] [security_policy <string>]
  ```

- The **`define | create space`** command has a new parameter to support index fields that set up distribution def fields:

  ```
  distribution_def ('KEY','field0','field1','field2' ...)
  ```

  Fields that are set up as distribution fields, will have their data stored on the same seeder if their field data is identical.

- The **`show | describe space`** command now indicates any fields that have been set up as distribution fields.

Release 2.0.2

With release 2.0.1, the default index type when adding primary keys was TREE.

Release 2.0.2 changes the default index type to HASH.

C Changes

- **Alter Space, Remove IndexDef** New functions have been added to alter the space definition for a space and add nullable fields, and add/remove index definitions after the space has been defined initially.
- **onAlter Method** `Persister.onAlter()` has been added to the `Persister` interface. This breaks backwards compatibility with current Persister implementations because `onAlter()` needs to be defined. `Persister.onAlter()` is called to signal changes to the space definition.
- **Invoke Options** The invoke APIs have been changed to accept an `InvokeOption`. This breaks backwards compatibility with previous callers as the function signature has changed.

Added:

- tibasSpaceDef_RemoveIndexDef()
- tibasMetaspace_AlterSpace()

Java Changes

The Java API has been changed to provide a `Metaspace.alterspace()` method.

Deprecated:

Invocable methods that take class names such as `InvokeSeeders()` and `InvokeLeeches()` have been deprecated.

- `InvokeResult invoke (Tuple keyTuple, String invocable, Tuple contextTuple) throws ASException.`
- `InvokeResult invoke (Tuple keyTuple, Class<? extends Invocable> cls, Tuple contextTuple) throws ASException;`
- `InvokeResult invokeMember (Member member, String invocable, Tuple contextTuple) throws ASException;`
- `InvokeResult invokeMember (Member member, Class<? extends MemberInvocable> cls, Tuple contextTuple) throws ASException;`
- `InvokeResultList invokeMembers (Collection<Member> members, String invocable, Tuple contextTuple) throws ASException;`
- `InvokeResultList invokeMembers (Collection<Member> members, Class<? extends MemberInvocable> cls, Tuple contextTuple) throws ASException;`

- `InvokeResultList invokeMembers (String invocable, Tuple contextTuple) throws ASException;`
- `InvokeResultList invokeMembers (Class<? extends MemberInvocable> cls, Tuple contextTuple) throws ASException;`
- `InvokeResultList invokeSeeders (String cls, Tuple contextTuple) throws ASException;`
- `InvokeResultList invokeSeeders (Class<? extends MemberInvocable> cls, Tuple contextTuple) throws ASException;`

Admin Language Changes

1. A new **alter space name** command has been added. The new command allows adding new fields to a space definition, adding a new index, and dropping an existing index.
2. The **show member** command and the **show space** command now display statistics for locks, unlocks, queries, and invokes.

Release 2.0.1

There are several changes to the API set in this release.

- The interface `InvocableMember` has been renamed to `MemberInvocable`.
- Use of the `disconnect()` method on the metaspace has been replaced with a `close()` method.
- Changed the name of the `getConnectionDef()` method of the metaspace interface to `getMemberDef()`.
- The methods of the `Invocable` and `MemberInvocable` interfaces have been modified to take an additional parameter to use for passing user defined context information.
- The `onJoin`, `onUpdate`, and `onLeave` methods of the `SpaceMemberListener` interface have been modified so that the space name is passed in along with the other method parameters.
- ActiveSpaces 2.0.1 fixes a defect that caused the primary key index (and all indexes) to be TREE indexes. The intent was to have the primary key index be a HASH index by default, and all other indexes be TREE indexes by default. Due to this fix (AS-919), key indexes are now HASH by default as intended.

- Added the following operation status settings:
  - MISMATCHED_TUPLE
  - MISMATCHED_LOCK
  - LIMIT_EXCEEDED
- The `SpaceState` enumeration now has two additional states: `WAITING` and `RECOVER`. The `FAILED` state has been removed from the `SpaceState` enum.
- Redistribution of data in the space is now done in a seamless manner.

  Previously, when data was being redistributed, operations on the Space would be blocked until the redistribution was completed. Now the redistribution is done gradually (in phases) so that operations will not notice that redistribution is occurring and can continue to function normally while the data is being redistributed. The user is allowed to specify the number of phases to use for redistributing the data and the amount of time between redistribution phases where normal operations can continue.
- Operations on a space will no longer immediately return an error if the space is not ready.
- A space wait time has been added to the space definition. Operations will now block for the specified space wait time waiting for the space to become ready again before returning an error.

C Changes

- **Space Wait Timeout, Read Timeout, Write Timeout** New functions have been added to set and get space wait timeouts, read timeouts, and write timeouts.
- **Context Functions** A group of context functions has been added to implement thread contexts that enable a thread to release locks and transactions and another thread to acquire the locks and transactions.

Added:

- tibasSpaceDef_GetSpaceWait()
- tibasSpaceDef_SetSpaceWait()
- tibasSpaceDef_GetWriteTimeout()
- tibasSpaceDef_SetWriteTimeout()
- tibasSpaceDef_GetReadTimeout()
- tibasSpaceDef_SetReadTimeout()
- tibasSpaceDef_GetLockScope()

- tibasSpaceDef_SetLockScope()
- tibasTuple_Clone()
- tibasMetaspace_ReleaseContext()
- tibasMetaspace_AcquireContext()
- tibasMemberDef_SetRemoteListen()
- tibasMetaspace_GetRemoteListen()
- tibasMetaspace_ListenMemberEvents()
- tibasMetaspace_ListenSpaceMemberEvents()
- tibasMetaspace_ListenSpaceDef()
- tibasMetaspace_ListenSpaceState()
- tibasMemberDef_SetContext()
- tibasMember_GetContext()
- tibasMemberDef_SetTimeout()
- tibasMember_GetId()
- tibasSpace_Invoke()
- tibasSpace_InvokeMember()
- tibasSpace_InvokeMembers()
- tibasSpace_InvokeRemoteMembers()
- tibasSpace_InvokeSeeders()
- tibasInvokeResult_GetMember()
- tibasSpaceMemberListener_Create())
- tibasRemoteSpaceMemberListener_Create()
- tibasMemberListener_Create()
- tibasRemoteMemberListener_Create()
- tibasSpaceStateListener_Create()
- tibasSpaceDefListener_Create()
- tibasMemberDef_SetWorkerThreadCount()
- tibasMemberDef_GetWorkerThreadCount()
- tibasMemberEvent_GetType()
- tibasMemberEvent_GetManagementRole()

- tibasMemberEvent_GetMember()
- tibasMemberEvent_Free()
- tibasSpaceMemberEvent_GetType()
- tibasSpaceMemberEvent_GetSpaceName()
- tibasSpaceMemberEvent_GetDistributionRole()
- tibasSpaceMemberEvent_GetMember()
- tibasSpaceMemberEvent_Free()
- tibasSpaceRemoteMemberEvent_GetType()
- tibasSpaceRemoteMemberEvent_GetSpaceName()
- tibasSpaceRemoteMemberEvent_GetDistributionRole()
- tibasSpaceRemoteMemberEvent_GetMember()
- tibasSpaceRemoteMemberEvent_Free()
- tibasRemoteMemberEvent_GetType()
- tibasRemoteMemberEvent_GetProxyMember()
- tibasRemoteMemberEvent_GetRemoteMember()
- tibasRemoteMemberEvent_Free()

Deprecated:

- tibasMetaspace_ReleaseTransaction()
- tibasMetaspace_TakeTransaction
- tibasSpaceMemberDef_Create()
- tibasSpaceMemberDef_SetDistributionRole()
- tibasSpaceMemberDef_SetDataStore()
- tibasSpaceMemberDef_Free()

.NET API Changes

The following changes have been made to the .NET API:

- Removed the file `SpaceEntry.cs` as it is no longer used.
- Removed the file `Types.cs` as it is no longer used.
- Added the class `Listener/SpaceStateListener`.

Release 2.0.0

There are a number of changes to the API in this release.

General Changes

- Space and browser functions now return tuples rather than entries.
- There are several new CLI commands and changes to existing commands. See Admin Language Changes, page 27
- Functions that previously returned TIBAS_NOT_FOUND now return NULL.
- Functions that previously took entries or entry lists as arguments now take tuples or tuple lists.
- Get, Put, Take, and Lock functions now have an “extended” version that allows you to specify lock wait time, lock, unlock, and forget values.

C Changes

- `tibasMetaspace_Connect()` now accepts a `memberDef` struct instead of a `tibasConnectionDef` struct.

Added:

- tibas_SetLogLevel()
- tibas_GetLogLevel()
- tibas_GetMetaspace()
- tibas_GetMetaspaceNames()
- tibasSpaceDef_SetKey()
- tibasSpaceDef_SetKeyDef()
- tibasSpaceDef_GetKeyDef()
- tibasSpaceDef_AddIndexDef()
- tibasSpaceDef_GetIndexDef()
- tibasKeyDef_Create()
- tibasKeyDef_GetIndexType()
- tibasKeyDef_SetIndexType()
- tibasKeyDef_GetFieldNames()
- tibasKeyDef_SetFieldNames()

- tibasKeyDef_Free()
- tibasIndexDef_Create()
- tibasIndexDef_GetName()
- tibasIndexDef_SetName()
- tibasIndexDef_GetIndexType()
- tibasIndexDef_SetIndexType()
- tibasIndexDef_GetFieldNames()
- tibasIndexDef_SetFieldNames()
- tibasIndexDef_Free()
- tibasIndexDefList_Size()
- tibasIndexDefList_Get()
- tibasIndexDefList_Free()
- tibasSpaceMemberDef_Create()
- tibasSpaceMemberDef_SetDataStore()
- tibasSpaceMemberDef_Free()
- tibasTuple_Clear()
- tibasSpaceResult_GetStatus()
- tibasSpaceResult_GetTuple()
- tibasSpaceResult_HasError()
- tibasSpaceResult_GetError()
- tibasSpaceResult_Free()
- tibasSpaceResultList_Put()
- tibasSpaceResultList_Size()
- tibasSpaceResultList_Get()
- tibasSpaceResultList_GetStatus()
- tibasSpaceResultList_GetTuple()
- tibasSpaceResultList_GetTuples()
- tibasSpaceResultList_GetError()
- tibasSpaceResultList_HasError()
- tibasSpaceResultList_Free()

- tibasMetaspace_GetSpaceEx()
- tibasMetaspace_RecoverSpace()
- tibasMetaspace_RecoverSpaceEx()
- tibasMetaspace_GetRemoteDiscovery()
- tibasMemberDef_Create()
- tibasMemberDef* memberDef)
- tibasMemberDef_Free()
- tibasMemberDef_SetMemberName()
- tibasMemberDef_SetListen()
- tibasMemberDef_SetDiscovery()
- tibasMemberDef_SetRemoteDiscovery()
- tibasMemberDef_GetMemberName()
- tibasMemberDef_GetListen()
- tibasMemberDef_GetRemoteDiscovery()
- tibasSpace_PutEx()
- tibasSpace_PutAllEx()
- tibasSpace_LoadEx()
- tibasSpace_LoadAllEx()
- tibasSpace_TakeEx()
- tibasSpace_TakeAllEx()
- tibasSpace_CompareAndPut()
- tibasSpace_CompareAndPutEx()
- tibasSpace_CompareAndPutAll()
- tibasSpace_CompareAndPutAllEx()
- tibasSpace_CompareAndTake()
- tibasSpace_CompareAndTakeEx()
- tibasSpace_CompareAndTakeAll()
- tibasSpace_CompareAndTakeAllEx()
- tibasSpaceDef_Create()
- tibasSpaceDef_AddIndex()

- tibasSpaceDef_GetIndex()
- tibasSpaceDef_SetPersistenceType()
- tibasSpaceDef_GetPersistenceType()
- tibasSpaceDef_SetPersistencePolicy()
- tibasSpaceDef_GetEvictionPolicy()
- tibasSpaceDef_SetLockNull()
- tibasSpaceEvent_HasOldTuple()
- tibasSpaceEvent_GetTuple()
- tibasSpaceEvent_GetOldTuple()
- tibasSpaceEvent_GetMetaspaceName()
- tibasSpaceEvent_GetSpaceName()

Deleted:

- tibasMetaspace_ConnectEx()
- tibasSpace_WaitForReadyEx()
- tibasSpace_PutAndLock()
- tibasSpace_PutAndLockAll()
- tibasSpace_Remove()
- tibasSpace_RemoveAll()
- tibasSpace_Update()
- tibasSpace_UpdateAll()
- tibasSpace_UpdateAndLock()
- tibasSpace_UpdateAndLockAll()
- tibasSpace_LockEntry()
- tibasSpace_LockAllEntries()
- tibasSpace_LockAllEntries()
- tibasSpaceDef_GetCapacityPolicy()
- tibasSpaceDef_GetCapacityPolicy
- tibasSpaceEvent_GetEntry()
- tibasSpaceEvent_IsNew()

- tibasSpaceDef_GetKey()
- tibasSpaceDef_AddIndex()
- tibasFieldDef_IsKey()
- tibasFieldDef_SetKey()
- tibasSpace_LoadEx()
- tibasSpace_LoadAllEx()
- tibasMemberDef_SetRemote
- tibasMemberDef_GetRemote

Java Changes

The Java API has been changed to provide class member functions equivalent to the new C API functions.

Admin Language Changes

1. The syntax for the `connect` command has been changed to include an optional `metaspace` parameter, which specifies the metaspace name to connect to.
2. The syntax for the `define | create space` command has been changed to include the following new arguments:
   - `key ( [type <string>] fields (<string> (, <string>)*))`
   - `(index ( name <string> [type <string>] fields (<string> (, <string>)*)))*`
   - `[persistence_type <string>]`
   - `[persistence_policy <string>]`
   - `[replication_policy <string>]`
   - `lock wait`
   - `lock scope`
   - `[lock_scope <string>]`
3. A new `recover space` command has been added to enable recovering a space. If a space goes into WAITING or INITIAL state when shared-nothing persistence is implemented, you can use this command to recover the space.
4. The `show| describe` space command now shows the new settings that can be enabled by the `define | create space` command.

Release 1.1.1

There are a number of changes to the API in this release.

General Changes

- `Unicast URL` is now referred to as `listen URL`, and `multicast URL` is now referred to as `discovery URL`.
- The `listen URL` port now defaults to `50,000`, instead of randomly choosing a port above `30,000`. If the port is already in use, the port number is automatically incremented to the first available port.

C Changes

- `tibasMetaspace_Connect()` now accepts `tibasConnectionDef` instead of separate multicast and unicast URL parameters.
- Added:

  ```
  tibasMetaspace_Disconnect()
  tibasConnectionDef_Create()
  tibasConnectionDef_Free()
  tibasConnectionDef_SetListen()
  tibasConnectionDef_GetListen()
  tibasConnectionDef_SetDiscovery()
  tibasConnectionDef_GetDiscovery()
  ```

Java Changes

- `Metaspace.connect()` now accepts `ConnectionDef` instead of separate multicast and unicast parameters.
- Added:

  ```
  ConnectionDef.create()
  ConnectionDef.setListen()
  ConnectionDef.getListen()
  ConnectionDef.setDiscovery()
  ConnectionDef.getDiscovery()
  ConnectionDef.setListen()
  ConnectionDef.getListen()
  ```

- Batch operations now return `SpaceResultList` instead of `ResultList`.
- `Collection<InvokeResult> InvokeResultList.getResults()` has been removed and replaced with `Collection<Tuple> InvokeResultList.getReturns()`.

Release 1.1.0

There are no changes in functionality in this release.

Release 1.0.2

This release includes the following changes.

Platform Support

- Added support for Solaris 10 (x86, x86-64)

C Changes

- The following C API functions now accept `tibasEntry` instead of `tibasEntry*`

  `tibasSpace_LockEntry`

  `tibasSpace_Unlock`

  `tibasEntry_Refresh`

- DateTime object type is added with following options

  `DateTime.create()`

  `DateTime.create(long )`

  `DateTime.create(Calendar)`

Release 1.0.1

There are a number of changes to the API in this release.

C Changes

1. Renamed object types so they now match function prefixes, for example, `tibas_tuple` is now `tibasTuple`. Renamed enumerated type `tibas_role` to `tibas_distributionRole`. Added a enumerated type `tibas_managementRole`.
2. Removed `tibasError_Next`. Instead, `tibasError_GetMessage` and `tibasError_GetStackTrace` should now be used.
3. Renamed `tibasListenerDef_SetInitialValue` to `tibasListenerDef_SetTimeScope`.

4. Added:

```
tibasMetaspace_GetMembers
tibasMetaspace_GetSpaceMembers
tibasMetaspace_GetUserSpaceNames

tibasMemberIter type
     tibasMemberIter_Next
     tibasMemberIter_HasNext
     tibasMemberIter_Free

tibasMember_GetManagementRole
tibasMember_GetDistributionRole

tibasStringIter type
     tibasStringIter_Next
     tibasStringIter_HasNext
     tibasStringIter_Free
```

5. `tibasDateTime` contains `sec` and `nsec`. These values used to be from Julian Epoch, but now they are from Posix (Unix) Epoch (1-1-1970), so now the results of Posix system calls like `gettimeofday` can be stored directly into `tibasDateTime` without being converted to Julian sec. Removed `julian2greg` and `greg2julian`.
6. Renamed `tibasTuple_Take` to `tibasTuple_Remove` to conform with Java `Tuple.remove`.
7. Renamed `tibasTuple_Merge` to `tibasTuple_PutAll`.
8. Removed `tibasMetaspace_GetSystemSpace` as `tibasMetaspace_GetSpace` now allows access to system Spaces.
9. The factory functions no longer exist. You can now create objects using the objects' `Create` functions. For example, `tibasFactory_CreateTuple` is now `tibasTuple_Create`.
10. The Metaspace object is now created using the `tibasMetaspace_Connect` function.
11. Removed `tibas_memberIterator`.
12. `tibasTuple_GetFieldNames` now returns a `tibasStringIter` object (as does the new `tibasMetaspace_GetUserSpaceNames` function).

13. Renamed:

    `tibas_fieldDefIterator` to `tibasFieldDefIter`

    `tibas_datetime` to `tibasDateTime`

    `tibasMetaspace_Commit` to `tibasMetaspace_CommitTransaction`

    `tibasMetaspace_Rollback` to `tibasMetaspace_RolllbackTransaction`

14. Changed `tibasDate_Free(void**)` to `tibas_FreeData(char**)`.
15. `tibasMember_GetName` and `tibasError_GetMessage` now returns internal strings that do not need to be explicitly freed.

Java Changes

1. The factory objects no longer exist. You can now create objects using the objects' static `create` method. For example, `Factory.createTuple` is now `Tuple.create`.
2. The Metaspace object is now created using the `Metaspace.connect` method.
3. DateTime functions now accept Java `Calendar` instead of the deprecated Java `Date` object.
4. `Tuple.put` methods now return old value or null instead of current Tuple.
5. Renamed `Tuple.merge` to `Tuple.putAll`. The method now returns void instead of the current Tuple.
6. Removed `Metaspace.getSystemSpace` as `Metaspace.getSpace` now allows access to system Spaces.
7. Replaced `Metaspace.leaveSpace` with `Space.leave`.
8. Packages have been rearranged.
9. `Metaspace.commit` has been renamed to `Metaspace.commitTransaction` and `Metaspace.rollback` has been renamed to `Metaspace.rollbackTransaction`.
10. Enum changes:
    - `Role` is changed to `DistributionRole`
    - `MetaspaceRole` is changed to `ManagementRole`
    - Enums such as `LogLevel`, `ListenerTimeScope`, `ListenerDistributionScope`, `BrowserTimeScope`, `BrowserDistributionScope`, `BrowserType`, and `FieldType` are moved to their own class files.

11. Added:
    - `Member.getDistributionRole`
    - `Member.getManagementRole`
12. `getUserSpaceNames` now returns `Collection<String>`.

Admin Language Changes

The syntax of the `connect` command has been changed. The following examples illustrate the new command syntax:

- `connect`
- `connect <metaspace_name>`
- `connect <metaspace_name> unicast <unicast_url>`
- `connect <metaspace_name> multicast <multicast_url>`
- `connect <metaspace_name> unicast <unicast_url> multicast <multicast_url>`
- `connect <metaspace_name> multicast <multicast_url> unicast <unicast_url>`

Packaging Changes

The `as-common.jar` file has been broken up into three separate files:

— **as-common.jar** Includes class files for all non-admin related API calls.

— **as-admin.jar** Includes class files used with Admin object and the Admin CLI tool.

— **antlr-3.2.jar** Third-party software required by the `as-admin.jar`.

Developers need to include the `as-admin.jar` file and the `antlr-3.2.jar` file in their CLASSPATH if they want to use the Admin object.

Release 1.0.0

There are no changes in functionality in this release.

Changes to the Example Code

The following modifications have been made to the example code provided with TIBCO ActiveSpaces.

New Features

Release 2.1.1

The sample programs have been updated to work with ActiveSpaces security.

AS Security Examples

The following new security-related examples were added in this release:

- examples/c/security/ASDomainController.c
- examples/c/security/ASUserAuthenticator.c
- examples/dotnet/security/ASDomainController.cs
- examples/dotnet/security/ASUserAuthenticator.cs
- examples/java/security/ASDomainController.java
- examples/java/security/ASUserAuthenticator.java

For documentation on the example security programs, see ASDomainController on page 183 and ASUserAuthenticator on page 186 in the Tutorial chapter of the *TIBCO ActiveSpaces Developer's Guide*.

AS Performance Examples

A set of ActiveSpaces performance programs has been added in the `/tools/asperf` subdirectory of each language set's example directory:

C Examples

- ASPerf.c
- ASPerfAgent.c
- ASPerfSlave.c

.NET Examples

- ASPerf.cs
- ASPerfAgent.cs

- ASPerfSlave.cs

Java Examples

- ASPerf.java
- ASPerfAgent.java
- ASPerfSlave.java

For documentation on the example performance programs, see ASPerf on page 189 in the Tutorial chapter of the *TIBCO ActiveSpaces Developer's Guide*.

Release 2.0.2

The sample programs have been updated to reflect any API changes for release 2.0.2.

Release 2.0.1

The following new C examples were added in this release:

- `InvokeClient`
- `MetaspaceMemberMonitor`
- `SpaceDefMonitor`
- `SpaceMemberMonitor`
- `SpaceStateMonitor`

The following new .NET examples were added in this release:

- `ASPaint`
- `MetaspaceMemberMonitor`
- `SpaceDefMonitor`
- `SpaceMemberMonitor`
- `SpaceStateMonitor`

The following additions were made to the Java examples in this release:

- `ASQuery` demonstrates how to compose filters to querying the entries of a space.
- `<AS_HOME>/examples/java/examples/persistence/MySQLConnection.java` was added to demonstrate connecting to a MySQL database when using shared-all persistence.

Changes in Functionality

Release 2.1.1

Release 2.1.1 includes the following functionality changes:

- The parameters for the following example programs were changed to allow operation with the ActiveSpaces security feature:
  - ASOperations
  - ASBatchOperations
  - ASChat
  - ASQuery
  - ASPersistence
  - ASRequestReplyServer
  - ASRequestReplyClient
  - InvokeClient
  - ASBrowser
  - ASEventBrowser
  - ASListener
  - MemberspaceMemberMonitor
  - SpaceDefMonitor
  - SpaceStateMonitor
  - SpaceMemberMonitor
- The Java and .NET `InvokeClient` examples were modified to match the C `InvokeClient` example.
- The namespace used for the .NET `InvocableLibrary` was changed to `remote` to allow the .NET `InvokeClient` example to be able to interoperate with the Java and C `InvokeClient` examples.
- .NET `ASInvocable` and `ASInvocableMember` are now linked with the .NET `ASOperations` example so that the .NET version of `ASOperations` can now be used with the `InvokeClient` example.
- Support for the **`-remote_discovery`** command line option has been fully removed from the examples. Remote discovery should now be specified using the **`-discovery`** command line option giving a value of the form:

  ```
  tcp://IP:port?remote=true
  ```

- The default timescope used in the `ListenDef` object for the Java and .NET examples is now "new_events". The Java, .NET, and C examples now all use the same default timescope setting.
- The Java, .NET and C examples all use a default timescope setting of SNAPSHOT in the BrowserDef object.

Release 2.0.1

The following modifications were made to all of the examples in this release:

- The examples for each API set were modified so that the behavior of each example would be consistent with the behavior of the same example written using a different API set.
- The examples for each API set were modified so that they would be able to interoperate with each other. For example, the Java version of `ASChat` can now be used with the .NET and C versions of `ASChat`.
- Unless an example requires a specific space definition, all of the examples were modified to take the following command line arguments for defining the characteristics of a space:
  - replication_count
  - capacity
  - eviction_policy
  - distribution_policy
  - min_seeders
- The `-timescope` command line argument was changed to `-time_scope`.
- The `-persistence` command line argument of the Java and .NET examples was modified to take the values `of shared_nothing` and `shared_all` instead of using `share_nothing` and `share_all`.
- A default space name of `shared_nothing_persisted` will be used when **`-persistence shared_nothing`** is entered on the command line.
- A default space name of `shared_all_persisted` will be used when **`-persistence shared_all`** is entered on the command line.
- The `SpaceMemberMonitor` and `SpaceStateMonitor` examples no longer require a space name to be specified. If the space name is null, all spaces in the metaspace will be monitored.
- The ASOperations example was updated so that it can be used to demonstrate shared-nothing persistence.

- The `ASOperations` example was updated so that it can act as a seeder for remote invocations.

The following modifications were made to the C examples in this release:

- On UNIX platforms, the file `Makefile.mk` now contains the platform specific makefile rules and is included by `Makefile`.
- On Windows, the file `Makefile` should be used for building the examples instead of Makefile.examples. Makefile now only contains those rules which are specific to building on Windows.
- All of the C examples were modified so that common code that reads in the command line arguments is located in the source file `<AS_HOME>/examples/c/base/ASExampleProps.c`.
- All of the C examples were modified so that common code that handles metaspaces, spaces, errors and object cleanup is located in the source file `<AS_HOME>/examples/c/base/ASExampleBase.c`.
- `ASOperations` was modified so that it can be used to demonstrate remote invocations in conjunction with the new `InvokeClient` example.

The following modifications were made to the .NET examples in this release:

- Moved `ASRequestReplyClient.cs` into the directory `<AS_HOME>/examples/dotnet/RequestReplyClient`
- - Moved `ASRequestReplyServer.cs` into the directory `<AS_HOME>/examples/dotnet/RequestReplyServer`
- Modified the remote invocation example to the use default space so that `ASOperations` can be used instead of requiring a separate `RemoteServer` example.

The following modifications were made to the Java examples in this release:

- `<AS_HOME>/examples/java/README.txt` has been modified to point users to Chapter 4 of the *TIBCO ActiveSpace Developer's Guide* for detailed information on how to build and run the examples.
- `<AS_HOME>/examples/java/build.xml` has been modified to build both `Examples.jar` and **`ASPaint.jar`**.

Release 2.0.1

The following have been deprecated from the examples in this release:

- Use of `<AS_HOME>/examples/c/Makefile.examples` has been deprecated. `<AS_HOME>/examples/c/Makefile` should be used instead.

- Use of `<AS_HOME>/examples/java/examples/ASPersistence2.java` has been deprecated as `ASOperations` can now be used to demonstrate shared-nothing persistence.
- Use of `<AS_HOME>/examples/Java/ASPaint/build.xml` has been deprecated.
- `<AS_HOME>/examples/Java/build.xml` now builds both Examples.jar and ASPaint.jar.
- Use of the remote server examples in `<AS_HOME>/examples/dotnet/Remote` and `<AS_HOME>/examples/dotnet/RemoteServer`.

Compatibility

This section lists information pertaining to compatibility of ActiveSpaces with certain software environments, such as virtualized environments.

Running ActiveSpaces on Virtual Machines

You can run TIBCO ActiveSpaces on virtual machines (VMs) in most environments.

Under some circumstances, issues have been observed:

1. IP multicast on VMs has unpredictable latency times, and sometimes drops data, which causes GMP multicast to fail or causes various issues.
2. VMWare snapshots might cause members to disconnect from each other.
3. Running on VMs in Linux 32-bit and 64-bit environments or Solaris X86 32-bit or 64 bit environments, multicast discovery sometimes fails or hangs. This occurs more often with TIBCO RV discovery than with PGM discovery.

Closed Issues

The following table lists issues that were closed in the named releases.

Closed in Release	Key	Summary
2.1.2 Add-On	AS-2541	ActiveSpaces now supports using authentication and identity parameters from admin scripts, the agent command line, and APIs, with secure metaspaces.
2.1.2 Add-On	AS-2698	Space size method with filter now throws an exception when issued from leech clients.
2.1.2 Add-On	AS-2701	ActiveSpaces now supports a write-behind mechanism for the shared-all persistence type.
2.1.2 Add-On	AS-2707	A default query limit of 10,000 queries is now implemented, to avoid cluster stability issues for snapshot queries.
2.1.2 Add-On	AS-2713	Member names of remote clients are now checked to make sure they are unique.
2.1.2 Add-On	AS-2730	Shared nothing recovery now occurs for all the spaces when recovery occurs.
2.1.2 Add-On	AS-2733	Proxy nodes no longer generate a “space_member_def_mismatch” error right after a remote client joins a cluster.
2.1.2 Add-On	AS-2758	The Space size operation now throws an exception if the space is not ready.
2.1.2 Add-On	AS-2825	ActiveSpaces now supports cross-site replication.
2.1.2 Add-On	AS-2807	A tibasMetaspace_CloseAll function has been added to the C API.
2.1.2	AS-2117	Gets performed during redistribution returned NULL for an existing entry.

Closed in Release	Key	Summary
2.1.2	AS-2118	Takes performed during redistribution returned NULL for an existing entry.
2.1.2	AS-2123	The as-admin utility crashed when all other nodes were killed one at a time.
2.1.2	AS-2120	Rolling upgrade from release 2.1.1 to release 2.1.2 does not work.
2.1.2	AS-2124	With replication enabled, more entries were being evicted from a space than indicated by the capacity setting for the space.
2.1.2	AS-2138	HP SSL issue: With HP SSL, the error: "map text for library</groupfiler/qaql/bhaldar/hpux/ita64/as/2.1/lib/libssl.so.1.0.0>: mmap(0x0, 0xdd020, 0x5, 0x41, 3, 0x0)" occurred, and a Permission denied message was returned.
2.1.2	AS-2146	Get browsers returned old values for entries taken when replication was enabled. This issue has been resolved.
2.1.2	AS-2152	Improve usage of shared-nothing persistence files. This has been resolved.
2.1.2	AS-2201	Throttling of operations for shared-nothing persistence should be improved. This issue has been resolved.
2.1.1	AS-312	A new thread was being created each time `Space.setPersister` was called. This issue has been fixed.
2.1.1	AS-1278	When a Get browser was implemented with a time scope of All, new entries added to the space were not counted. This issue has been fixed.
2.1.1	AS-1325	C API: Missing tibasSpace_GetMetaspace() and tibasSpace_GetMetaspaceName().

Closed in Release	Key	Summary
2.1.1	AS-1430	Java ASPerf master and slave show different results for the average time when latency put or latency take is done.
2.1.1	AS-1481	Add JavaDoc description for `MemberDef.setDataStore`. `MemberDef.java` now has complete JavaDoc.
2.1.1	AS-1496	ActiveSpaces should have a JDBC driver example.
2.1.1	AS-1603	2.1.1 `as-agent.jar`: Command line parsing did not catch invalid command args using _ instead of -
2.1.1	AS-1714	Use of **`-remote_discovery`** should be removed from the examples.
2.1.1	AS-1715	Missing .NET version of `ASPerf`. A .NET version has been added.
2.1.1	AS-1759	If the Discovery/Listen URL contained extra spaces or newline characters, it was is considered as an invalid URL.
2.1.1	AS-1947	A system crash occurred when queries were made using `mod()`, `time()`, `date()`, `year()` and related signatures. This issue has been fixed.
2.1.1	AS-1954	A system crash occurred when multi-threaded transport was used. This issue has been fixed.
2.1.1	AS-1955	Disconnect from a space hung because worker threads did not stop.
2.1.1	AS-1972	When the logging level was set to `TIBAS_LOG_INFO` (the default logging level), and persistence was used, a debug message indicating <Route> was output. This issue has been fixed.

Closed in Release	Key	Summary
2.1.1	AS-1974	With security authentication enabled on the Linux platform, the as-admin utility did not accept a password; therefore, it was not possible to connect to a security domain requestor. This issue has been fixed.
2.1.1	AS-1976	On the Solaris platform, it was not possible to connect to a security domain requestor using LDAP authentication.
2.1.1	AS-1977	PGM discovery did not work on HP-UX 32-bit systems. This issue has been fixed.
2.1.1	AS-1992	For tuples larger than 20 KB, file space re-use was deteriorating and fragmentation was increasing. This issue has been fixed.
2.1.1	AS-2003	Java `InvokeClient` appeared to be hung on doing an invoke on the key when encryption was enabled.
2.1.1	AS-2012	C `InvokeClient` did not work on members, seeders, or self, remote when run with the C `ASOperations` example.
2.1.1	AS-2014	.NET InvokeClient example could not interoperate with the Java or C version of the same example.
2.1.1	AS-2015	.NET ASOperations could not be used with `InvokeClient` example.
2.1.1	AS-2017	No tibasMemberDef_SetDataStore() function was available in the C API. This issue has been fixed.
2.1.1	AS-2038	The Alter Space operation allowed setting of `spaceDef` attributes, causing a system crash. This issue has been resolved.
2.1.1	AS-2041	A system crash occurred when a secondary hash index was used. This issue has been fixed.

Closed in Release	Key	Summary
2.1.1	AS-2047	The linker generated an Unresolved External Symbol error for `tibasInvokeResultList_GetResults()`. This function has been renamed to `tibasInvokeResultList_GetReturns()` to match the function names in the Java and .NET API sets, and the linker issue has been resolved.
2.0.2 HF-03	AS-2037	The as-admin utility crashed when disconnecting from a metaspace.
2.0.2 HF-03	AS-2061	Using the Datetime field as a key caused an issue in distribution.
2.0.2 HF-03	AS-2089	The as-admin utility crashed when all other nodes were killed one at a time.
2.0.2 HF-03	AS-2111	When running BW as a seeder, BW would become a leech intermittently. Upon losing the seeder, the space changed to the RECOVER state if there were not enough seeders as defined in the SpaceDef for the space.
2.0.2 HF-03	AS-2115	[SN+TX] With shared-nothing persistence enabled, if the client was killed right after a commit was called, the committed entries were present in the space but not in the shared-nothing persistence data store file.
2.0.2 HF-03	AS-2119	Takes performed during redistribution returned NULL for an existing entry.
2.0.2 HF-03	AS-2121	Gets performed during redistribution returned NULL for an existing entry
2.0.2 HF-03	AS-2128	High CPU usage was observed on nodes processing get operation.
2.0.2 HF-03	AS-2143	Throttling of operations for shared-nothing persistence should be improved. This issue has been resolved.

Closed in Release	Key	Summary
2.0.2 HF-03	AS-2144	Improve usage of shared-nothing persistence files. This has been resolved.
2.0.2 HF-02	AS-1662	A datetime value in tuples caused the system to go down.
2.0.2 HF-02	AS-1673	Data loss occurred with repeated space recovery.
2.0.2 HF-02	AS-1675	The ActiveSpaces browser was not returning all data values.
2.0.2 HF-02	AS-1690	Spaces remained in the READY state after a persister error occurred, causing creation of duplicate records.
2.0.2 HF-02	AS-1692	Two entries were stored in the shared nothing file for puts when shared-nothing persistence was used with transactions.
2.0.2 HF-02	AS-1680	When an alter space operation was performed, a newly added index could be used before it was initialized fully.
2.0.2 HF-02	AS-1775	ActiveSpaces sometimes crashed with share-nothing persistence if the same key was written at the same time via a batch or concurrent operation.
2.0.2 HF-02	AS-1778	The invokeMember method crashed for a Java Native Interface (JNI) method and resulted in a stack trace being printed.
2.0.1 HF-01	AS-1420	When using unicast discovery, dropping all discovery nodes from a metaspace and bringing any back up resulted in a split-brain condition that was not always recoverable. This issue has been fixed.

Closed in Release	Key	Summary
2.0.2 HF-01	AS-1788	A memory leak was observed for the following API calls for ActiveSpaces 2.0.2: • Metaspace.getSpaceDef • Browser routines • Metaspace.browse • Space.browse • Listener routines • Metaspace.listen • Space.listen The memory growth was slow but happened each time one of these methods was called. `As-admin` was using a browser to retrieve statistics, so the issue also applied to `as-admin` statistics.
2.0.2 HF-01	AS-1662	A datetime value in tuples caused the system to go down. This issue has been fixed.
2.0.2 HF-01	AS-1673	Data loss occurred with repeated space recovery. This issue has been fixed.
2.0.2 HF-01	AS-1675	The ActiveSpaces browser was not returning all data values. This issue has been fixed.
2.0.2 HF-01	AS-1690	Spaces remained in the READY state after a persister error occurred, causing creation of duplicate records. This issue has been fixed.
2.0.2 HF-01	AS-1692	Two entries were stored in the shared nothing file for puts when shared nothing persistence was used with transactions. This issue has been fixed.
2.0.2 HF-01	AS-1680	When an alter space operation was performed, a newly added index could be used before it was initialized fully. This issue has been fixed.

Closed in Release	Key	Summary
2.0.2 HF-01	AS-1775	ActiveSpaces sometimes crashed with share-nothing persistence if the same key was written at the same time via a batch or concurrent operation. This issue has been fixed.
2.0.2 HF-01	AS-1778	The invokeMember method crashed for a Java Native Interface (JNI) method and resulted in a stack trace being printed.
2.0.2	AS-125	Float and tuple values were truncated. This issue has been fixed.
2.0.2	AS-579	Starting multiple leech members in quick succession was causing the system to crash. This issue has been fixed.
2.0.2	AS-916	When a member left or was dropped from a space, all locks set by remote clients were cleared, which made locks set by remote clients unreliable. This issue has been fixed.
2.0.2	AS-919	Setting the index type in IndexDef and KeyDef had no effect. A TREE index would always be created. This issue has been fixed.
2.0.2	AS-929	Under certain circumstances, using a Browser or Listener in a Remote Client would crash the client. This issue has been fixed.
2.0.2	AS-1099	The JavaDoc for ActiveSpaces did not include several methods defined in the Java API set. This issue has been fixed.
2.0.2	AS-1356	Define Space was failing silently. This issue has been fixed.

Closed in Release	Key	Summary
2.0.2	AS-1378	When running `ASOperations` and a remote invocation occur ed on it, `ASOperations` did not exit completely, but remained hung after the space and metaspace were closed. The same occurred with remote/InvokeClient. If a user chose the 'self' menu option so that the remote invocation occur ed on remote/InvokeClient itself, remote/InvokeClient would not exit. This issue has been fixed.
2.0.2	AS-1379	ActiveSpaces DATETIME was losing milliseconds. This issue has been fixed.
2.0.2	AS-1410	The C example `makefile` gave an error when **`/usr/ccs/bin/make`** was used. This issue has been fixed.
2.0.2	AS-1494	On HP-UX 11.31, using CIM and Weblogic, Java was not able to find libstd_v2.so, a system library. This issue has been fixed.
2.0.2	AS-1506	Leech applications were using excessive CPU resources on HP-UX shcedidv B.11.23 U ia64. This issue has been fixed.
2.0.2	AS-1598	Unable to recover all spaces when ActiveSpaces is restarted. This issue has been fixed.
2.0.2	AS-1171	Redistribution on member join, leave, and drop was blocking normal operations until complete. This issue has been fixed.
2.0.1 HF-01	AS-1488	The ActiveSpaces timer manager caused high CPU usage and context switching when a large number of spaces were defined (200 or more spaces). This was fixed by reducing the timer manager scan interval.
2.0.1 HF-01	AS-1508	On AIX, due to a defect in SpinLock, ActiveSpaces crashed intermittently at startup. This issue has been fixed.

Closed in Release	Key	Summary
2.0.1 HF-01	AS-1509	Queries on hashed indexes returned inconsistent results. Index selection was sometimes incorrect and queries provided either incorrect results or resulted in processing of queries for longer than necessary. General indexing issues have been addressed in Hotfix 1, and the specified issues have been fixed.
2.0.1 HF-01	AS-1532	ActiveSpaces 2.0.1-HF1 is compatible with BusinessEvents 5.1.0-HF1.
2.0.1	AS-908	When the ASMM UI received an update from the server, the ordering of members and space names was not always consistent. This resulted in rows moving around in tables and charts changing colors that were previously associated with another member/space. This issue has been fixed.
2.0.1	AS-893	When using remote client space operations that accept options, the options were ignored. This issue has been fixed.
2.0.1	AS-915	All locks set by remote clients were cleared when any member that was not a remote client left or dropped. This issue has been fixed.
2.0.1	AS-917	Hash indexes did not work correctly. This issue has been fixed.
2.0.1	AS-926	Browser/listener based tests failed under some circumstances when used with a remote client. This issue has been fixed.
2.0.1	AS-1224	UTF-8 was not supported in Java. ActiveSpaces now supports UTF-8.

Closed in Release	Key	Summary
2.0.0 HF-01	AS-909	Under certain circumstances, when using all members, when persisters were killed, the console would print an unnecessary protocol timeout warning, even though the request had properly returned with an error. This issue has been fixed.
2.0.0 HF-01	AS-911	Under certain circumstances, when using shared nothing persistence, protocol timeout messages were seen on the client. This issue has been fixed.
2.0.0 HF-01	AS-938	Unable to store any byte array into a tuple BLOB field. This issue has been fixed.
2.0.0 HF-01	AS-1077	If the operating system did not have TCP keepalive enabled, connections between members would close after a long period of inactivity, causing a member to drop. This issue has been fixed.
2.0.0 HF-01	AS-1079	Indexes were not returning correct results when composite indexes were used. This issue has been fixed.
2.0.0 HF-01	AS-1080	Snapshot browsers were hanging indefinitely when many browsers were created over a short time frame in the Linux environment. This issue has been fixed.
2.0.0 HF-01	AS-1081	During a transaction, when a seeder joined or left and caused a redistribution of tuples, the transaction state was not being redistributed correctly, causing an inconsistency of transaction state. This could cause the transaction to appear committed to clients reading the uncommitted changes. This issue has been fixed.
2.0.0 HF-01	AS-1083	Browsers, event browsers, and listeners would sometimes miss some tuples or events related to them if the tuples were involved in a transaction. This issue has been fixed.

Closed in Release	Key	Summary
2.0.0 HF-01	AS-1085	If a Take occurred within a transaction, the Take was being immediately committed instead of waiting for the commit or rollback. This issue has been fixed.
2.0.0 HF-01	AS-1086	When recovering a space using shared-all persistence with replication, seeders would crash. This issue has been fixed.
2.0.0 HF-01	AS-1089	Browsers, event browsers, and listeners created using the .NET API were not releasing metaspace resources when they closed. This issue has been fixed.
2.0.0 HF-01	AS-1090	Using the .NET API with the 32-bit library would always throw an exception on a metaspace connect. The .NET version of as-agent would error out. This issue has been fixed.
2.0.0 HF-01	AS-1091	Calling Space.isReady() in the .NET API was causing a crash. This issue has been fixed.

Known Issues

The table in this section lists known issues related to TIBCO ActiveSpaces.

Defect #	Summary/Workaround
AS-910	**Summary:** Cannot drop a space definition unless all members have left the space **Workaround:** All members should leave the space before dropping the space definition
AS-912	**Summary:** On 32-bit Windows platforms, when members of the metaspace run out of memory, the process may exit with no error message. **Workaround:** Deploy more members to distribute load on memory.
AS-1113	**Summary:** With shared-nothing persistence, space recovery fails when there is a partially generated data store file and a complete data store file. **Workaround:** None.
AS-1124	**Summary:** When the number of seeders goes below the min-seeder count, rollback doesn't work properly. **Workaround:** Deploy enough seeders to meet the min-seeder count.
AS-1426	**Summary:** Redistribution of nondistributed spaces blocks all write operations until redistribution is complete. **Workaround:** None.
AS-1529	**Summary:** When multiple instances of space object perform a space.close() operation more than one time, this causes an exception. **Workaround:** None.
AS-1683	**Summary:** Recovery with shared-nothing persistence depends on node startup order. This results in only some nodes recovering if a user is performing recovery without all nodes (that were part of the cluster previously). **Workaround:** Issue the **`recover`** command after all of the nodes have joined the space.

Defect #	Summary/Workaround
AS-1727	**Summary:** `OPERATION_TIMEOUT (protocol_timeout)` exceptions were observed when system memory became low. There are additional situations when a protocol_timeout exception might occur: • Bringing up a new seeder while puts are occurring. Due to redistribution, put operations might time out, because writes are not given priority during redistribution. **Workaround:** Increase the write timeout value using the `MemberDef.setWriteTimeout()` function. • Under Low memory conditions, where operations slow down, resulting in a protocol_timeout exceptio **Workaround:** Increase available memory. Increasing the write timeout value will not help; it will just delay the problem. • With shared-all persistence: If shared-all persistence operations take a long time due to the system of record being slow (for example, when the database is located at a remote site) write operations might time out. **Workaround:** Increase the write timeout value using the. `MemberDef.setWriteTimeout()` function.
AS-1975	**Summary:** Calls to the `tibasTuple_PutDateTime()` function crash if user sets a `tibasDateTime.sec` value prior to the EPOCH time. **Workaround:** Use the `LONG` field type and use fields defined with that field type to store datetime information.
AS-2004	A `MemberManager::processMemberHeartbeatTimeout` exception occurs in the following scenario: • Two Java clients are started, one as a seeder and one as a controller. • A large number of puts is done on each of the clients. • While the puts are in progress, a Java client is started with the ASOperations sample program as a requestor and a size operation is performed. **Workaround:** None.

Defect #	Summary/Workaround
AS-2050	**Summary:** With Solaris Sparc x86, the uninstaller may find as-common.lib locked and fail. **Workaround:** Enter the following command on the command line to disable file locks, and then proceed with the uninstallation: `-V disableLockChecks="true"`
AS-2172	**Summary:** Running ActiveSpaces with shared-nothing persistence with capacity set can cause a secondary index to be removed. This causes queries to fail using the secondary index **Workaround:** None.
AS-2194	**Summary:** When shared-nothing persistence is used and capacity is set, Take browsers are not able to take all entries on mixed platforms such as Solaris and AIX. **Workaround:** None.
AS-2202	**Summary:** On Solaris, queries cause a crash for the following examples: (ANNUAL_SALARY + 10000.00) BETWEEN 1000000.00 and 1022000.00 ANNUAL_SALARY * 2 > 3004000.00 abs(-1)-abs(1) **Workaround:** None.