

# White Paper

**bi-Cube<sup>®</sup> SSO**

**Experiential Report of a Qualified  
Single Sign-On**

Technologies Solutions Trends Experience



## Table of contents

<b>1</b>	<b>SUMMARY .....</b>	<b>3</b>
<b>2</b>	<b>QUALIFIED SINGLE SIGN-ON; REQUIREMENTS AND APPLICATION SCENARIOS .....</b>	<b>3</b>
2.1	Intrinsic Safety .....	4
2.2	Availability .....	5
2.3	Gina Integration .....	5
<b>3</b>	<b>DOES SSO LEAD TO ANOTHER SECURITY PROBLEM?.....</b>	<b>6</b>
3.1	Dual Authentication .....	6
3.2	Active Authentication .....	6
3.3	Four-eye Principle with Biometrics .....	7
<b>4</b>	<b>BENEFITS OF A Q-SSO SOLUTION .....</b>	<b>7</b>
4.1	Increased Convenience for the User .....	7
4.2	Cost Reduction, most of all for the User Help Desk (UHD) .....	7
4.3	Significant Increase of Security .....	7
<b>5</b>	<b>IMPORTANT FUNCTIONS OF Q-SSO.....</b>	<b>8</b>
5.1	Q-SSO-API .....	9
5.2	Logon Technologies.....	9
<b>6</b>	<b>PASSWORD MANAGEMENT .....</b>	<b>10</b>
6.1	Methods of Password Management .....	10
6.2	Password Synchronization and Q-SSO Cluster .....	11
6.3	Q-SSO Version Cluster.....	11
<b>7</b>	<b>Q-SSO VERSUS FEDERATED IDENTITY .....</b>	<b>12</b>

## 1 Summary

This article summarizes the experiences of several SSO projects and highlights, in particular the requirements which must be followed when an SSO project is to be successfully in complex IT structures (and only these are dependent on SSO solutions).

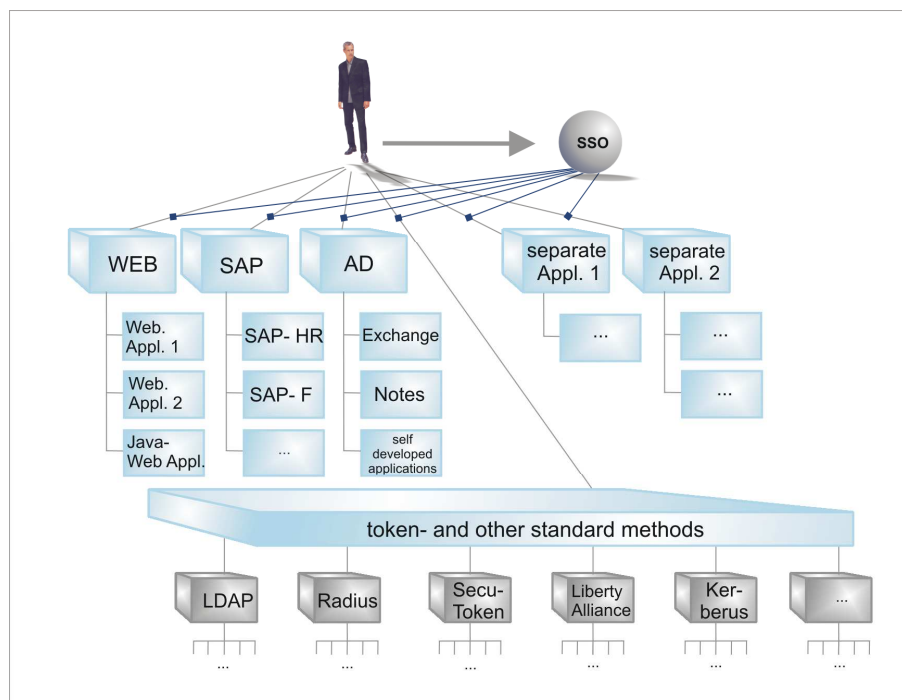
These requirements lead to a new category of SSO solutions which are here referred to as a qualified SSO (Q-SSO).

**Each potential user of an SSO solution should clarify whether a singular SSO solution is sufficient, or if perhaps a Q-SSO is required for the given conditions.**

## 2 Qualified Single Sign-On; Requirements and Application Scenarios

The need to concern oneself with the subject of Single Sign-On results on the aspects of security requirements for the use of different systems and the application of own authorization management.

To access a large quantity of systems of many of today's companies, the user must remember on regular basis 3 to 6 different logon procedures with sophisticated parameters (user ID and password) and apply to the rules and regulations. Since this can be very complicated, the user searches for support which usually violates the security policies.



A qualified SSO combines or optimizes the different options of account management with the goal to provide an easy logon for the user. At the same time the IT system security is increased. The Q-SSO assumes the initial logon for the user to the sub systems for which no other system's password-synchronization or identity transfer exists.

**Picture 1:** Position of the Q-SSO in combination with other Identity-Management options

Another negative result of this system environment is determined at the UHD (user help desk) each Monday or after a longer period of absence, (e.g. vacation or illness) when the number of password-reset requests did increase.

IT-admins often promise a planned portal development with which everything supposed to be resolved in one step. But even if this aspect is realization (when ever), more logons or authentications (LAN, e-mail, not portal compatible applications etc.) remain. Q-SSO is a wise addition even if a portal is the target architecture.

The introduced requirements are related to Q-SSO, which is to increase the security as well as the usability of the IT recourse with diverse cross-platform added functions. The concrete solution clearly stands above other trials which offer, by this acronym, additional functions of standard systems (SAP, Microsoft, Novell etc.); but often only implement a synchronization of accounts within the platform.

Furthermore Q-SSO fulfils diverse other requirements and additional functions.

In a heterogeneous environment of LAN to Basis Win2000 / AD and NT to UNIX and Host, possibly all essential applications should be implemented in a Q-SSO desktop. "Essential" are those applications, available to at least 80 % of users. Or special programs as an expert system that are used by only a few users (Insurance Co.) to valuate automotive damage. This is usually not included, since only about 10 users use this application. Access to the damage-incentive system, used by many insurance clerks is therefore imperative to SSO systems.

This system must operate in a terminal server environment (e.g. Citrix), it must include functions of an Application Launcher, ICS (Internal control system)-function, the four-eye principle and realize diverse additional functions, e.g. quick user change, file encryption and the implementation of a personalized Q-SSO.

It must enable password synchronization in a mixed LAN world (NT incl. ADS /win2000). Such a mixed LAN world is regularly found in large companies, whose migration to ADS/win 2000 covers a longer period. Users are managed in the AD system but the file spaces are still partially implemented in a Network environment.

Besides the primary (central) profile, the users may create their own personalized profiles with Q-SSO. Subordinate stored systems which are not relevant to a SSO can be imported by the user individually.

The Q-SSO functionality must also be available even if the corresponding client is not connected with the Q-SSO server or the Internet. The field staff logs on via the Internet. The Q-SSO client must recognize each system environment and displays them separately.

Of course Q-SSO must have a high intrinsic safety and availability.

## 2.1 Intrinsic Safety

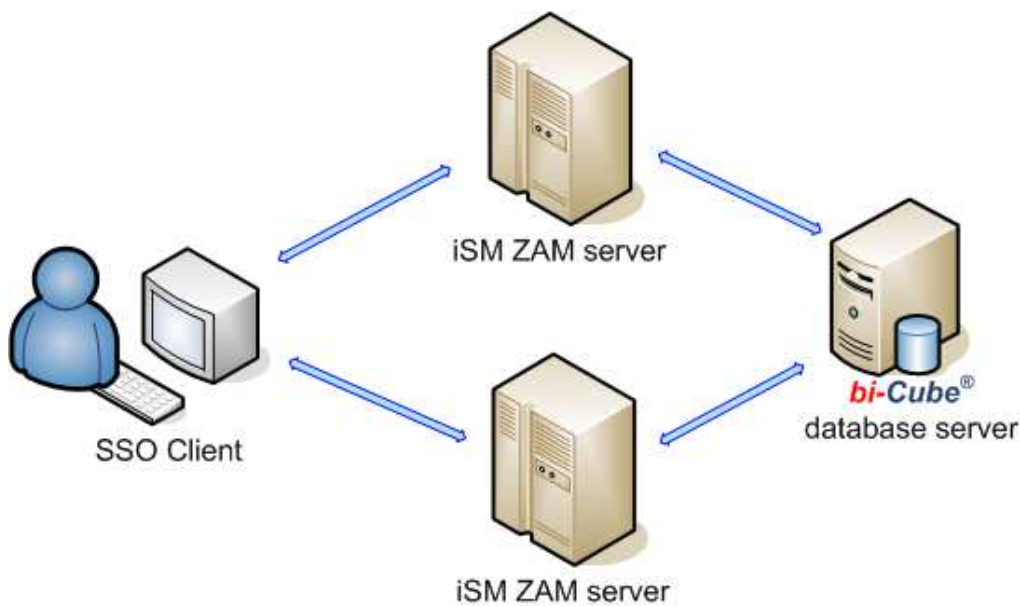
The entire Q-SSO system, beginning with the rights management of the administration, the transfer of logon data to the client, up to the storage to the client must be under a complete security concept.

## 2.2 Availability

The availability of Q-SSO functionality has become even more significant considering the fact that users cannot know their password because it was automatically generated by SSO or due to the use of the SSO he has simply forgotten his password.

It is achieved by clustering the Q-SSO server via a load balancing and a local saving of the Q-SSO profile. This must be combined with a highly stable Q-SSO client.

Availability criterion is also the correct functionality of the Q-SSO client. Meaning, the Q-SSO must always be able to correctly perform the logon instead of the user. A qualified password management is a problem to be solved. However it must be solved for each target system individually. (see also Password Management)



**Picture 2:** Load balancing and applications cluster of a Q-SSO

## 2.3 Gina Integration

The functions of the GINA (Graphical Identification and Authentication) and of the SSO optimally co-ordinate with each other in a Q-SSO. This is why a Q-SSO usually is available with its own integrated Gina. This in particular is important for the elimination of the below-described security issues.

### 3 Does SSO lead to another Security Problem?

A SSO solution is often rejected since it may generate a new security problem.

A SSO solution is often rejected since it may generate a new security problem. At different accounts (user ID and password) the risk of a "cracked" access is usually limited to one concrete system and not all systems at the same time. A Q-SSO system offers an adequate solution! This solution consists of the following components:

#### 3.1 Dual Authentication

For a Q-SSO a safe access to the SSO-Client is created. The different steps are summarized in the term "highly secure authentication". This includes all types of dual authentications, especially in conjunction with biometrics.

#### 3.2 Active Authentication

The biometrics has the advantage of an active authentication. At any time the direct presence of the user is demanded, which is not so with some other dual authentication procedures. A dual procedure is not proficient if it can be activated at absence of the user (e.g. query of a memory card).

The active authentication in the secure Q-SSO should be activated through modeling "prior" to each specially saved application. This means that the launch of such an application in the **bi-Cube<sup>®</sup> SSO-Client** first requires the active authentication of the user. The user must at the start be present in person and must either conduct a biometric authentication or enter his LAN password before the actual logon begins.

This procedure must also be applied at the SSO configuration.  
(see picture)



**Picture 3:** Biometric authentication via Q-SSO, prior to the start of an application which is to be backed up.

### 3.3 Four-eye Principle with Biometrics

With the use of biometrics and Q-SSO technologies, requirements can be added after additional release during a business process (e.g. release of a large payment) without intervening to the application. The releasing person uses a biometric device which then determines in **bi-Cube<sup>®</sup>** the password for the approval. This password is, with the logon technology Q-SSO, written to the logon input field without the releasing person needing to know the password.

## 4 Benefits of a Q-SSO Solution

### 4.1 Increased Convenience for the User

This effect proves to be undisputed and is, with a secure functioned SSO, greatly accepted by the user.

### 4.2 Cost Reduction, most of all for the User Help Desk (UHD)

This factor is offered as a primary effect by all providers. The indication of cost reduction at UHD differs between 30% and 60%. With Q-SSO an essential reduction in UHD-calls is attained. This effect is greatly enhanced with the combination of Q-SSO and IPM (IPM = Identity and Provisioning Management). This includes mainly:

- procedure of password self service
- refresh, triggered by the server of local profiles
- no details of the user at software assignment (start account)

User's losses of working hours, due to failed system access, are eliminated with Q-SSO. This also improves the availability of the overall IT infrastructure.

Q-SSO can also significantly reduce the license cost.

### 4.3 Significant Increase of Security

A qualified SSO (including the above aspects of intrinsic safety, etc. taken into account) leads to a great increase in security levels.

- The user does not have to violate against password rules.
- The policy of passwords for each system can be set high, since the user no longer needs to memorize them.
- The generally known security gap (at delivering initial-passwords via e-mail or phone) is completely eliminated.

## 5 Important Functions of Q-SSO

Besides the mentioned functional requirements (e.g. mixed environment at Citrix), the following functions are included with Q-SSO:

### 1. Quick User Change

The waiting period of a quick user change (log on, log off) to the operating system can be avoided. A selected person has to log on to the SSO-Client and a not personified user logs on to the operating system. It is required that the application does not need a personal directory. This necessity arises for example when used in hospitals and in banks.

### 2. Extended Log Ability via the Q-SSO Event Management

With the extended Log it is possible to log the start of the SSO-Client to the database and the activation of applications included in the SSO-Client. The data records are listed and analyzed by the report and can be used for e.g. License Optimization.

The SSO-Client offers the following functions for the Event-Management:

1. Logging of Q-SSO logon or log off
2. Logging of logon periods to a system (for License Optimization)

The events can be activated depending of the analyzed target:

- Launching the Q-SSO-Clients
- Exit the Q-SSO-Clients
- Update the Q-SSO-Clients
- Launching an application via the Q-SSO

Standard evaluation can be executed as follows:

- Analyze by number of user / user which did not use the system more than x days.
- Average utilization rate (x times in 30 days) of a system and its roles

### 3. Reaction of LAN-availability

The availability of the Server is tested when the SSO-Client is launched. When the server is not available a query follows which application profiles (personal or local) are to be used. This test is repeated at manual update and if necessary the online condition is restored.

### 4. Reaction of Network Speed

If the network speed is reduced (e.g. at VPN), the time consuming actions like, test of existence/files or reading of icons, is skipped.

### 5. Automatic Update of the Application Profile (Trigger)

When the profile in the IPM database is changed, a message is sent to the SSO-Client. The SSO-Client then updates the application profiled. If the SSO-Client does not run at the moment, this message is earmarked. This prevents long security transfers of start accounts to the user.

### 6. User Self Registration

The **User Self Registration** (USR) makes it possible for the user to transfer an existing account to the Q-SSO system. This procedure may be used for mild accounts migration into the Q-SSO system. This is often the only option to make a password of a user's application known to the Q-SSO system.

The user selects the system to which he has access to and which is managed by the Q-SSO. By entering the User ID and password, the Q-SSO then automatically executes a logon for the user. The Q-SSO system then checks if the logon was successful and the logon data is set to "valid" at the Q-SSO server and therefore ready to be used.

## 7. Hidden SSO

With a so called "**Hidden SSO**", required authentications can be implemented to an application. Not only the start of an application is supported by an automatic logon, also required authentications within the application are executed. Quite often this could be of use in some applications (e.g. e-Bay). With Lotus Notes the user may lock his desktop by pressing F5 and unlock it again by entering a password. But usually this password has been forgotten so the Q-SSO assumes this part.

## 8. Variations with Different Operating Systems

All operating system-related input strings of a Q-SSO system must be available to the client. Only here it is decided which operating system the client includes. Thus, the SSO Client must be able to differentiate on what Windows version it currently runs. This is in contrast to program versions which require different SSO functionalities and which can be already selected at the server.

## 9. Additional Optional Function of a Q-SSO

- Different validity of passwords to each system
- Revocation of system assignment if the system was not entered within x days at Q-SSO level.
- System specific validity of the local logon without connection to the server (important for the field staff)
- System specific indication whether a system should be used via remote access.
- System specific indication whether a system can be used in Q-SSO by autorun.
- With a valid from and valid until date for each system a time can be specified from when and until when a system is available in Q-SSO, independent of the system assignment. With this a system can be made available for all users with only one entry of the test phase. The access can be enabled and disabled.
- Different user actions can trigger an alert to each e-mail address defined to a corresponding system.
- For each system it can be specified, from which system the password is "inherited".
- Separation of systems according to system environment (Application Launcher)
- 

## 5.1 Q-SSO-API

For important control and data transfer operations, Q-SSO provides an API to effectively integrate the system into existing management systems. This is particularly necessary or useful if the Q-SSO is not integrated into an IPM and the provisioning systems of other manufacturers or company specific management systems are used.

## 5.2 Logon Technologies

In order to integrate a broad palette of applications, various technologies are provided to implement an automatic interaction of the SSO system to the corresponding application. This is called horizontal integration. The Q-SSO system should provide the following minimum application technologies to the administrator:

### Security Token

The Security-Token (also called: certificate) is an easy and secure method of automated logon of the user to a system with its own authorizations management. The SSO-Client generates a one time password which includes other parameters (encrypted) like, User ID, time stamp and system ID of the activating system. This token is only valid 5 minutes (standard) and is fragmented to its individual parameters and checked by the target system. The token is rejected after the validity has expired or if the system is not known to the target system.

### Logon via API

This technology of automated user logon is safe and perhaps the one to choose. The application offers a program interface which enables the user logon as well as to manage logon data in the target system. Typical examples are SAP and 3270.

### User Logon Simulation

For this procedure, Windows functions are used to import required data directly into input windows. This process requires a more or less complicated scripting, reliable identification of the windows which expect an input and an ingenious time control (How long does the Q-SSO client wait for a window to appear, or starting when is the wait period defined so that the application is not started.)

Applications with different titles and multi step logon procedures complicate this process considerably. The creation of the scripts should definitely be supported by a scanner which co-writes the logon to the target system and automatically generates the script and saves it in the database to the system data.

A separate scanner is required for the Internet browser. This scanner must also support several display technologies (HTML, Java, etc.) of the respective logon GUI. The simulation of the keyboard is a "sub-type" of this technology which is used if other procedures are not applicable.

## 6 Password Management

The main quality of a Q-SSO is the Password Management which is needed because some users do not remember their password. The Password Management is useful if the password in SSO and the password in the target system do not match. It is possible that the user has changed his password in the target system and in this case should be able to also change the password in SSO.

**SSO-Synchronization** allows the user to pass on his changed password to the SSO. The last solution is to reset the password through the SSO functionality.

The Password Management is also required for systems which command a password change and the function cannot be ignored or canceled. As far as possible the Password Management is supported by Q-SSO. Thus the Q-SSO takes over the user's password change (in defined time periods).

The password management is marked primarily through the specificity of each system, and thereby quite complex in terms of a comprehensive implementation. Q-SSO offers a wide range of technical options for the password management.

### 6.1 Methods of Password Management

#### Manual Change (Method 0)

The user himself changes his password in the application client of the target system. If the password change was successful, the new password must be registered to **bi-Cube<sup>®</sup>** via the SSO-Client. This method might not be very practical but is available for each application.

### **Manual Password Change with Automated Registration to IPM (Method 1)**

The SSO-Client recognizes the appearance of the password change window and automatically saves the user's entry (the new password) and registers the new password in **bi-Cube<sup>®</sup>**.

### **Decentralized Password Change (Method 2)**

The SSO-Client monitors the password validity. In case a password is about to expire, the SSO-Client triggers the password change. A new password is created and the user's action is simulated in the application client of the target system.

### **Decentralized Password Change via API (Method 3)**

This method is similar to method 2, but here a new password is entered via the API of the application. The API offers the most security. To be allowed to execute the function of the password change, sometimes the client must communicate with the application in a system-context.

### **Central Password Change (Method 4)**

The password is changed by the central Password Manager. The change in the target application is executed via the message room and an adequate output-connector. After confirmation by the program, the new password is written to **bi-Cube<sup>®</sup>** or set to "valid" and can therefore be used by the SSO-Client. Usually it is not required to inform the user about the change.

Required is an output-connector for the target application, which is able to execute a password change. The validity in **bi-Cube<sup>®</sup>** IPM must be set, so that the Password Manager reacts prior to the password expiration or the validity is canceled in the application.

This method characterizes a Q-SSO because it is safe user-friendly. It is executed whether the user is logged on or not. However this option is only possible in systems with an integrated IPM-SSO architecture approach. Currently this is offered only by iSM's **bi-Cube<sup>®</sup>** solution.

## **6.2 Password Synchronization and Q-SSO Cluster**

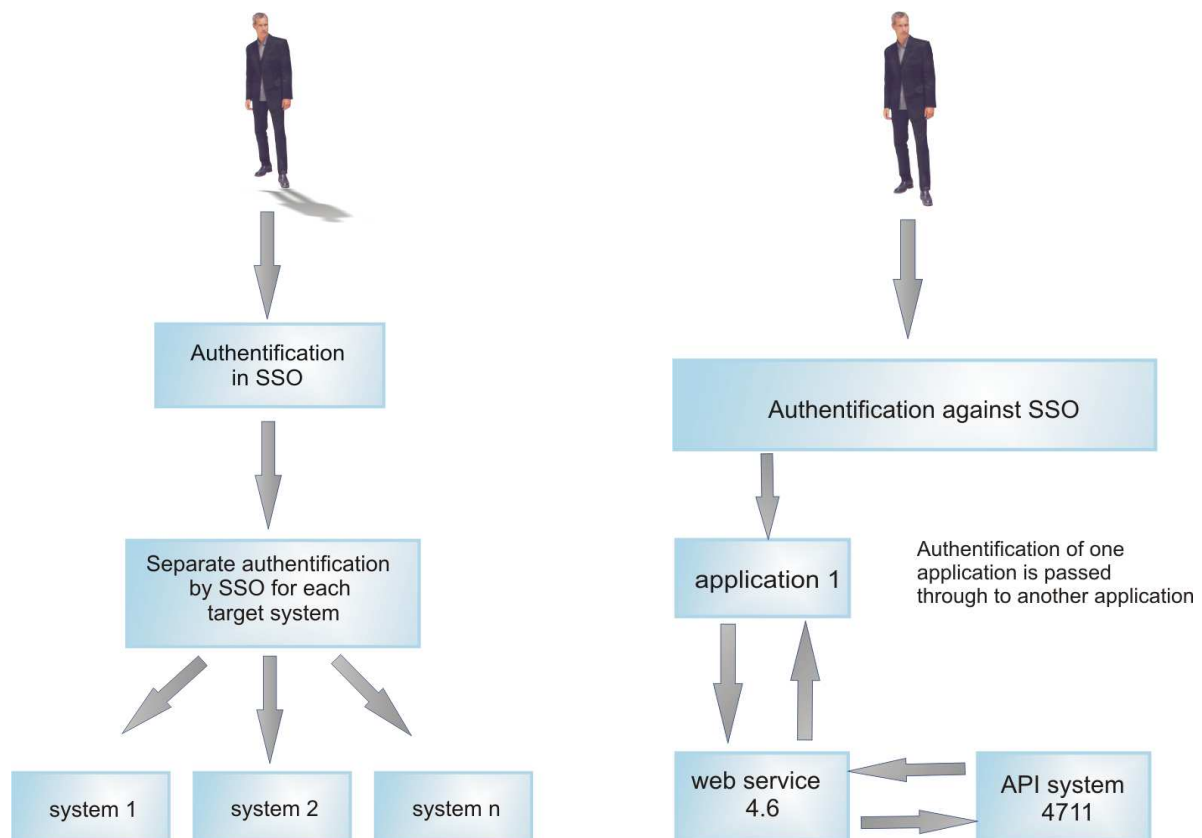
General password synchronization instead of the Q-SSO solution is basically inadequate due to security reasons. Within the Q-SSO solution, system clusters can be built, in between which a password synchronization (e.g. between all components of a product family, subject to the same password rules) can be conducted.

## **6.3 Q-SSO Version Cluster**

A special variant of a Q-SSO Cluster is when one application can be managed and assigned in various versions. One example is Lotus Notes which exists in several releases and versions of different languages and can be assigned to the user. This can only be controlled by a close integration of SSO with the IPM solution. Lotus Notes is defined as system and the versions / variants are defined as applications in the IPM. The account is linked to the system and the password distribution complies with the application. The modeling as a cluster always ensures the correct logon and allows the version change without changing the account data.

## 7 Q-SSO Versus Federated Identity

An SSO solution is basically a temporary solution for the lack of unitary architecture concepts, which provide a unique and standardized authentication of **all** applications. (see picture 1, below). For certain areas and application groups (e.g. SAP) a centralized identity (internally) is provided and used by all applications who acknowledge this identity. At the level above, always a range of other applications remain (at least for a foreseeable time) which are consolidated by SSO only for logon. From this perspective two options and partially also two objectives for SSO and Federated Identity remain.



**Picture 4:** Comparison of the function of SSO and Federated Identity