THE TICK42 CRM DICTIONARY

Incorporating the Tick42 Tracking Emails API

DRAFT VERSION

Abstract

The CRM dictionary defines a number of Glue methods that may be used to interact with a CRM database. The Tracking emails API extends these to support the requirements of recording and auditing email conversations using the CRM.

Dan Pike

danny.pike@tick42.com

DOCUMENT HISTORY

Version	Description	Author	Date
0.3	Initial version.	T Doust	9 March 2016
0.4	Bringing everything up to date with the latest demos.	A Tennant	18 March 2016
0.5	Updating to reflect requirements for Netsuite and RelPro integration	A Lancaster	21 February 2017
0.6	Tracking emails	D Pike	27 February 2017
0.7	Incorporated Tracking Emails and reworked the APIs to be more self-consistent and pulled the CRM dictionary into this document.	D. Pike	27th February 2017 – 8th March 2017
0.8	Several changes to the CRM API to simplify the structures and improve the naming conventions	D. Pike	27th March 2017
0.9	Specify return values for Create/Update/Delete methods. Extend contact and account definitions with context.	I. Pidov	28th April 2017

Contents

1 Introduction

This document defines the Glue methods used to allow a CRM system to work with a Helper system to look up and update details on Contacts (people), Accounts (companies) and Emails (tracking).

THE CRM METHODS

2 Method groups

The CRM methods described in this document are grouped into the following types:

- View-synchronisation methods.
- Side-by-side methods
- Inline methods (CRUD-based APIs).
- Tracking methods.
- Other methods.

This section summarises the methods in each group and is followed by a section that defines the methods individually. For each method, a "direction" is provided that indicates whether it is expected that the CRM expects the method to have been implemented by the Helper, or vice-versa. In some cases, the method may be implemented by both CRM and Helper.

2.1 View synchronisation methods

Glue can be used to select view synchronization so that when accounts or contacts are selected in one window, windows from other applications can update to show details for the selected account or contact.

Name	Direction	Description
T42.CRM.SyncAccount	**Both**	**Changes the information displayed in a window to match the account that is displayed elsewhere, keeping multiple windows synchronized with the account that they are showing.**
T42.CRM.SyncContact	**Both**	**Changes the information displayed in a window to match the contact that is being displayed elsewhere, keeping multiple windows synchronized with the account that they are showing.**

2.2 Side-by-side methods

Glue allows existing UIs from the CRM and Helper to work together as a hybrid application that combines the features of both into a single "user experience". This description assumes that that a Glue-enabled CRM window is visible, as is a Glue-enabled Helper window.

For the Account- and Context-related methods, a typical user story might be:

1. The user clicks a button in the CRM UI, possibly labelled "lookup details", which invokes the appropriate **T42.CRM.Resolve*** method published by the Helper.
2. The Helper handles the **T42.CRM.Resolve*** method, and displays a window showing all of the possible matching accounts/contacts. The CRM may send

additional data to the Helper, if the account/contact has been resolved beforehand, so that it may further refine the search parameters.

3. The user selects the correct account/contact in the Helper's window and clicks a button, possibly labelled "update details". This causes the Helper to invoke the appropriate **T42.CRM.Update*** method that is published by the CRM.

Name	Direction	Description
T42.CRM.ResolveAccount	**CRM to Helper**	**An account definition exists in the CRM and the user wishes to update the CRM with more complete or up-to-date data that is stored within the Helper.**
T42.CRM.ResolveContact	**CRM to Helper**	**The CRM requests the Helper to update the definition of a Contact, possibly prompting the user to resolve any ambiguities. Please refer to the description of T42.CRM.ResolveAccount for a more detailed description of the resolve-update cycle.**
T42.CRM.ResolveAccountContactList	**CRM to Helper**	**The user views the contact list for an Account in the CRM and is able to send this list to the Helper. The user can then update or add contacts as required.**

2.2.1 Resolving entities

The Tick42 CRM dictionary supports a mechanism for updating the CRM with new data stored in the Helper, where the data currently stored in the CRM is incomplete or out-of-date. This mechanism allows for the possibility that the CRM entity may potentially match more than one entity in the Helper, by allowing the following sequence ("user experience"):

1. The user selects the CRM entity and clicks a button to "ask" the Helper, by invoking the appropriate its **T42.CRM.Resolve*** methods.
2. The Helper's implementation of the **T42.CRM.Resolve*** method searches for and displays a set of equivalent entities in the Helper to the user. It will typically use a "fuzzy matching algorithm", as the CRM data may be an inaacurate "guess" as to the correct value. For example, it may not have been clear how to spell the name of the entity.
3. The user selects the correct value in the Helper and clicks a button to send the full definition of that entity data back to the CRM. The handler for this button in the Helper then invokes appropriate **T42.CRM.Update*** method implemented by the CRM, and passes the full and corrected definitinon to the CRM.
4. The CRM's implementation of the **T42.CRM.Update*** method corrects its database, thereby "resolving" the ambiguous or incomplete account definition that was previsouly stored.

2.3 Tracking methods

The **T42.CRM.LogContactCall** method is implemented by the CRM and may be invoked to enter details of a voice conversation with an existing client. The **T42.CRM.*Email** methods perform a similar function, but are intended to be used for email conversations, possibly involving several exchanges and a changing list of participants. The **T42.CRM.GetAttachment** method may be used by the CRM to retrieve the contents of any attachment to the email.

Name	Direction	Description
T42.CRM.LogContactCall	**Helper to CRM**	**The Helper sends a summary of a phone call to the CRM, so that it may be tracked.**
T42.CRM.TrackEmail	**Helper to CRM**	**The Helper sends an email to the CRM, so that it may be tracked.**
T42.CRM.UntrackEmail	**Helper to CRM**	**The Helper tells the CRM to remove an email, thereby "untracking" it.**

2.4 Inline methods (CRUD)

The following Glue methods allow the Helper to update account and contact data in the CRM, by using a create-update-delete workflow (CRUD).

Name	Direction	Description
T42.CRM.CreateAccount	**Helper to CRM**	**An account exists in the Helper. The user clicks a button and the account is added to the CRM.**
T42.CRM.UpdateAccount	**Helper to CRM**	**Transmits a "resolved" account definition from the Helper to the CRM. This may be used in conjunction with T42.CRM.ResolveAccount or unsolicited as part of a CRUD work cycle.**
T42.CRM.DeleteAccount	**Helper to CRM**	**An account exists in the Helper and, possibly, in the CRM. The user deletes the Account in the Helper and the Helper uses this method to delete the account from the CRM.**
T42.CRM.CreateContact	**Helper to CRM**	**A contact exists for an account in the Helper but not the CRM. The user is able to click 'export' and the contact is added to the account in the CRM.**
T42.CRM.UpdateContact	**Helper to CRM**	**Transmits a new contact definition from the Helper to the CRM. This may be used in conjunction with T42.CRM.ResolveContact or unsolicited as part of a CRUD work cycle.**

T42.CRM.DeleteContact	**Helper to CRM**	**The helper requests the CRM to delete a contact, if it exists.**

2.5 A note on the various Glue method types

Synchronisation methods are typically invoked as broadcast methods (also known as "all" methods). This is so that all interested applications can adjust to the new synchronisation state.

Any application that needs to both register and invoke one of these synchronisation methods must take care not to cause a race condition by inadvertently reacting and re-signalling its own notification. The other method types are typically point-to-point ("best" or "any") methods. Only two parties are involved and no particular attention need be paid to avoiding such a race condition.

Methods may be implemented as synchronous or asynchronous calls. In general, it is up to the programmer to decide which implementation is most appropriate but, it is recommended that any method that might involve some direct user interaction be implemented as an *asynchronous* Glue method. This will avoid unexpected errors from the call timing out, when the user fails to respond quickly enough.

3 Methods

This sections describes each of the methods in the groups listed above, defining the parameters that each method takes, any values or side-effects that the method may have on the CRM or the Helper, and whether it should have a *Glue entity tag*

Glue Entity Tags

These are strings that are used to indicate to the user-interface that this method is also intended to be used from a dynamically-constructed list of methods, e.g. a right-click context menu. The value of the tag describes the action that the method may have on the currently selected object.

For example, consider an application that produces one or more tables of data in a standard format. That application might implement a context menu that allows the user to request that the application forward copies of the table to one or more other processes. Using entity tags, it is possible for the first application to auto-detect that the other processes are running, and to include them all in its context menu, without having each of those processes written into its code.

The application should simply declare in its specification that its context menu will invoke any method that matches the given entity tag with data in the format that it chooses. Any other process can then publish a method with that same entity tag, and the application will add that method to its context menu. When the user selects the option that she wants, the application is able to invoke the correct method on the correct process without having to worry about where that process is running and how it will act upon the data once it is received.

3.1 T42.CRM.CreateContact

This method is used by the Helper to create a new contact in the CRM, typically in response to the user clicking a "Create Contact" button.

The CRM publishes this method to be invoked by the Helper.

3.1.1 Glue Entity Tag

T42.CRM.CreateContact does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.1.2 Parameters

Name	Type	Description
contact	**T42Contact**	**The definition of the new contact, which must not already exist within the CRM.**

3.1.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

Successful invocations **MUST** return the created contact object extended with the new contact id from the target system.

A typical reason for failure might be that a Contact already exists within the CRM, as decided by the CRM's entry in the **T42Contact**'s id field.

3.2 T42.CRM.DeleteContact

This method is used by the Helper to remove a contact from the CRM, typically in response to the user clicking a "Delete Contact" button.

The CRM publishes this method to be invoked by the Helper.

3.2.1 Glue Entity Tag

T42.CRM.DeleteContact does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.2.2 Parameters

Name	Type	Description
contact	**T42Contact**	**Identifies the contact to be deleted, which may or may not already exist within the CRM. It is up to the implementation of the CRM to define which properties of the T42Contact must be included for this method to succeed.**

3.2.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

Successful invocations **MUST** return an empty object.

This method is idempotent, in that it is not considered to be an error if the **T42Contact** does not exist within the CRM database.

3.3 T42.CRM.UpdateContact

This method is used by the CRM to ask a Helper to update its definition of a **T42Contact**. Please refer to section 2.2.1 "Resolving entities" on page 8 for a description of algorithm that is supported by **T42.CRM.UpdateContact** and its partner **T42.CRM.ResolveContact**.

3.3.1 Glue Entity Tag

T42.CRM.UpdateContact does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.3.2 Parameters

Name	Type	Description
contact	**T42Contact**	**The new definition of the contact**
autoCreate	**Bool?**	**Whether to create the contact in the Helper, if one does not already exist. The default is false.**

3.3.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

Successful invocations **MUST** return the updated contact object.

A typical reason for failure might be that the contact was not found and the **autoCreate** parameter was not set to **true**.

3.4 T42.CRM.ResolveContact

The CRM requests the Helper to update the definition of a **T42Contact,** possibly prompting the user to resolve any ambiguities. Please refer to section 2.2.1 "Resolving entities" on page 8 for a more detailed description of the **T42.CRM.ResolveContact** method and its parnet **T42.CRM.UpdateContact**.

3.4.1 Glue Entity Tag

T42.CRM.ResolveContact does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.4.2 Parameters

Name	Type	Description
contact	**T42Contact**	**The "incomplete" contact definition as it curently exists in the CRM**

3.4.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values. This might happen if, for example, the contact was not found.

3.5 T42.CRM.ResolveAccountContactList

This method is implemented by the Helper to allow user to update the contacts that are associated with a particular account.This is a one-way notification method; there is no corresponding **T42.CRM.ResolveAccountContactList** method, as for the other methods described in section 2.2.1. The Helper should use the CRUD methds summarised in section 2.4 "Inline methods (CRUD)" on page 9 to update the CRM.

3.5.1 Glue Entity Tag

T42.CRM.ResolveAccountContactList does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.5.2 Parameters

Name	Type	Description
account	**T42Account**	**The account that contains the list of contacts to be edited.**
contacts	**T42Contact[]**	**The list of contacts that are currently associated with the account.**

3.5.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values. This might happen if, for example, the contact was not found.

3.6 T42.CRM.CreateAccount

This method is used by the Helper to create a new account in the CRM, typically in response to the user clicking a "Create Account" button.

The CRM publishes this method to be invoked by the Helper.

3.6.1 Glue Entity Tag

T42.CRM.CreateAccount does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.6.2 Parameters

Name	Type	Description
account	**T42Account**	**The definition of the new account, which must not already exist within the CRM.**

3.6.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

Successful invocations **MUST** return the created account object extended with the new account id from the target system.

A typical reason for failure might be that the account already exists within the CRM. The algorithm used to detect duplicate accounts is implementation-specific and therefore outside the scope of this document.

3.7 T42.CRM.DeleteAccount

This method is used by the Helper to remove an account from the CRM, typically in response to the user clicking a "Delete Account" button.

The CRM publishes this method to be invoked by the Helper.

3.7.1 Glue Entity Tag

T42.CRM.DeleteAccount does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.7.2 Parameters

Name	Type	Description
account	**T42Account**	**Identifies the account to be deleted, which may or may not already exist within the CRM. It is up to the implementation of the CRM to define which properties of T42Account are required to identify the account that is to be deleted.**

3.7.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

Successful invocations **MUST** return an empty object.

This method is idempotent, in that it is not considered to be an error if the **T42Account** does not exist within the CRM database.

3.8 T42.CRM.UpdateAccount

This method is used by the CRM to ask a Helper to update its definition of a **T42Account**. Please refer to section 2.2.1 "Resolving entities" on page 8 for a description of algorithm that is supported by **T42.CRM.UpdateAccount** and its partner **T42.CRM.ResolveAccount**.

3.8.1 Glue Entity Tag

T42.CRM. UpdateAccount does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.8.2 Parameters

Name	Type	Description
account	**T42Account**	**The new definition of the account**
autoCreate	**Bool?**	**Whether to create the account in the Helper, if one does not already exist. The default is false.**

3.8.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

Successful invocations **MUST** return the updated account object.

Typical reason for failure might be that the account was not found and the **autoCreate** parameter was not set to **true**.

3.9 T42.CRM.ResolveAccount

The CRM requests the Helper to update the definition of a **T42Contact**, possibly prompting the user to resolve any ambiguities. Please refer to section 2.2.1 "Resolving entities" on page 8 for a

description of algorithm that is supported by **T42.CRM. ResolveAccount** and its partner **T42.CRM.UpdateAccount**.

3.9.1 Glue Entity Tag

T42.CRM.ResolveAccount does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.9.2 Parameters

Name	Type	Description
account	**T42Account**	**The "incomplete" account definition as it currently exists in the CRM.**

3.9.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values. This might happen if, for example, the account was not found.

3.10 T42.CRM.SyncAccount

Updates a "listening" application (Helper or CRM) so that it is processing (typically displaying) the same account as the application (Helper or CRM) that invoked the method. If the account does not exist in the "listening" application, that application should cancel the current selection, perhaps displaying a blank screen or a suitable message to the user. It must not return an error in this case.

3.10.1 Glue Entity Tag

T42.CRM.SyncAccount does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.10.2 Parameters

Name	Type	Description
account	**T42Account**	**The new account to select.**

3.10.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

3.11 T42.CRM.SyncContact

Updates a "listening" application (Helper or CRM) so that it is processing (typically displaying) the same contact as the application (Helper or CRM) that invoked the method. If the contact does not exist in the "listening" application, that application should cancel the current selection, perhaps displaying a blank screen or a suitable message to the user. It must not return an error in this case.

3.11.1 Glue Entity Tag

T42.CRM.SyncContact does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.11.2 Parameters

Name	Type	Description
contact	**T42Contact**	**The new contact to select.**

3.11.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

3.12 T42.CRM.LogContactCall

Requests that an application should record the contents of a call received from a contact. For example, a CRM might display a form, fill in some of the fields from the parameters supplied and allow the user to enter details of the call before storing it in the database.

3.12.1 Glue Entity Tag

T42.CRM.LogContactCall does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

3.12.2 Parameters

Name	Type	Description
contact	**T42Contact**	**The primary contact for the call.**
participants	**T42Contact[]?**	**Any other participants, i.e. when this is a conference call.**
subject	**String?**	**A summary of the purpose of the call.**

dueDate	**DateTime?**	**A follow-up deadline for an action to be created.**
description	**String?**	**Additional call notes**

3.12.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

4 Parameter Types

4.1 This section describes the composite parameter types that are used by each of the CRM methods described in section 2.5 "A note on the various Glue method types

Synchronisation methods are typically invoked as broadcast methods (also known as "all" methods). This is so that all interested applications can adjust to the new synchronisation state.

Any application that needs to both register and invoke one of these synchronisation methods must take care not to cause a race condition by inadvertently reacting and re-signalling its own notification. The other method types are typically point-to-point ("best" or "any") methods. Only two parties are involved and no particular attention need be paid to avoiding such a race condition. Methods may be implemented as synchronous or asynchronous calls. In general, it is up to the programmer to decide which implementation is most appropriate but, it is recommended that any method that might involve some direct user interaction be implemented as an *asynchronous* Glue method. This will avoid unexpected errors from the call timing out, when the user fails to respond quickly enough.
Methods" starting on page 10.

4.2 T42Id

This is the universal identifier class for an object in the CRM-Helper system. It consists of two elements:

(1) an optional user-friendly display name, that has no meaning to the CRM or the Helper but may be useful for displaying to the user; and
(2) an array of application-specific name-value pairs that identify the object to each window in the system.

Once a name-value pair has been added by a window, every other window must guarantee to continue to pass it to subsequent methods within the **T42Id** object. A window should not make any assumption about which of the pairs is meaningful to any other window in the system.

4.2.1 Members

Name	Type	Description

nativeId	**String**	**is the opaque value that identifies the object to the named system (CRM or Helper)**
systemName	**String**	**identifies the creator of the id ("Outlook", "MyCRM", "MyHelper", etc.)**

4.3 T42Name

This is a composite that defines a name. It is typically used as a member of another parameter type (T42Contact etc). The name should include at least one of the firstName or lastName fields. Typical values might be "Smith | John | Piers Algernon | Rt Hon | MBE PhD"

Name	Type	Description
companyName	**String?**	**In the case where this is a company, not a person**
lastName	**String?**	
firstName	**String?**	
otherNames	**String[]?**	**These are the middle names, excluding the firstName and the lastName.**
honorific	**String?**	**For example "Mr.", "Mrs.", Dr." etc.**
postNominalLetters	**String[]?**	**Qualifications such as PhD, MP etc.**

4.4 T42Contact

This is a Glue composite value that describes a single contact within the system.

Name	Type	Description
ids	**T42Id[]**	**Identifier for the contact; globally unique to the system**
displayName	**String?**	**Advisory only. A user-friendly name for this object to use in debug logs etc.**
name	**T42Name?**	**The name of the contact. Optional in case the name has not yet been resolved from the email, or id etc**
status	**String?**	**The status of the contact, e.g. "client", "prospect", "lead"**

isPerson	**Bool?**	**Is the contact a single person?**
account	**T42Id?**	**The account allocated to this contact**
addresses	**T42Address[]?**	**Each T42Address includes a description saying the type of address, e.g. "home", "office".**
phones	**T42Phone[]?**	**An array of phone numbers for this contact.**
emails	**String[]?**	
context	**Composite{}?**	**A dictionary of CRM specific fields**

4.5 T42Account

This is a Glue composite value that describes a single account within the system.

Name	Type	Description
ids	**T42Id[]**	**Identifier for the account; globally unique to the system.**
displayName	**String?**	**Advisory only. A user-friendly name for this object to use in debug logs etc.**
owner	**T42Id?**	**The account Id of the owning company, when this account is a subsidiary.**
subsidiaries	**T42Id[]?**	**Optional additional account Ids that are subsidiaries of this account.**
name	**String?**	**Name of the account, which may or may not be the name of a company / person.**
status	**String?**	**The status of the account, .e.g "lead", "prospect", "client".**
contacts	**T42Id[]?**	**A collection of key contacts within the account, e.g. "sales manager", "lead source"**
addresses	**T42Address[]?**	**Each T42Address includes a description saying the type of address, e.g. "home", "office".**
phones	**Composite[]?**	**An array of name-value pairs formatted as:** **{** **string number,** **string name,** **}**

		where: **number** **is the phone number** **name** **is a description, e.g. "home", "office"**
websites	**String[]?**	
emailDomains	**String[]?**	
billingEmailAddress	**String?**	**The billing email address for the account. For individual email addresses, refe to the T42Contacts that are associated with this account.**
Context	**Composite{}?**	**A dictionary of CRM specific fields**

4.6 T42Address

This is a composite that defines an address. It is typically used as a member of another parameter type (T42Contact etc).

Name	Type	Description
streetAddress	**String?**	
city	**String?**	
stateOrRegion	**String?**	
postalCode	**String?**	
country	**String?**	
description	**String?**	**The type of address, e.g. "home", "office"**

4.7 T42Phone

A phone number for a contact. There may be multiple phone numbers for each.

4.7.1 Members

Name	Type	Description

number	**String**	**The phone number**
name	**String**	**Identifies the phone number, e.g. "office", "home", "fax".**

THE TRACKING EMAIL API

5 Tracking Methods

This section is a continuation of the above, but describing the Tracking methods that apply to emails separately.

5.1 T42.CRM.TrackEmail

This method is used by the Helper to transmit an email to the CRM so that it may be tracked for auditing and/or reviewing the history of conversations between parties.

5.1.1 Glue Entity Tag

T42.CRM.TrackEmail does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

5.1.2 Parameters

Name	Type	Description
email	**T42Email**	**The email to be tracked**
conversationIds	**T42Id[]?**	**A sequence of emails may be related, e.g. when it one a reply to an earlier email. The conversationId is the correlationId for all emails that belong to the same "send-reply-reply" thread.** **If no conversationId is supplied, then the system will be unable to relate this email with any other in the system.**
forceNew	**bool?**	**When present and set to true, this tells the CRM to start a new conversation with this email, even if the conversationId in the email member matches an existing conversation stored in the CRM database.**
callback	**string**	**The name of the method to be invoked by the CRM when the tracking process is completed (success or failure)**
cookie	**string**	**The cookie is passed to the callback method to allow the Helper to retrieve contextual information about the email being tracked.**

5.1.3 Return Value

The CRM may choose to open a confirmation dialog for the user to add extra information to the email before it is added to the database, e.g. to convert the sender and recipient email addresses into internal codes. For this reason, **T42.CRM.TrackEmail** uses an explicit callback method to notify the Helper when the tracking process is complete, or if it has been cancelled.

The immediate return method from the function indicates only that the CRM has received the email from the Helper; it is not a confirmation that the email has been tracked. This confirmation only arrives by way of the callback method.

If the CRM were to complete the tracking process synchronously, it must still invoke the callback method. Depending on the threading model of the Helper, this may mean that the callback is invoked before the T42.CRM.TrackEmail returns any result, and the Helper must be coded with this possibility in mind.

5.1.3.1 Callback method

The callback method includes the following parameters:

Name	Type	Description
emailIds	**T42Id[]**	**The identifier of the email that the Helper asked to be tracked by the CRM. This is the same value that was passed in the id field of the T42Email parameter to the T42.CRM.TrackEmail method. The Helper may choose to use this as a correlation value in addition to or instead of the cookie.**
conversationIds	**T42Id[]**	**A sequence of emails may be related, e.g. when it one a reply to an earlier email. The conversationId is the correlationId for all emails that belong to the same "send-reply-reply" thread.** **If no conversationId is supplied, then the system wil be unable to relate this email with any other in the system.**
success	**bool**	**True if the email was tracked by the CRM. Otherwise an error occurred or the user cancelled the operation**
errorMessage	**String?**	**An optional error message describing the reason why the email was not tracked. This is used for logging and, depending on the Helper, it may be displayed to the user.**
cookie	**String**	**The cookie value that was passed to the T42.CRM.TrackEmail method by the Helper.**

5.2 T42.CRM.UntrackEmail

This method is used by the Helper to ask the CRM to untrack an email that has previously been tracked.

5.2.1 Glue Entity Tag

T42.CRM.UntrackEmail does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

5.2.2 Parameters

Name	Type	Description
emailIds	**T42Id[]**	**The email to be untracked**
conversationIds	**T42Id[]?**	**A sequence of emails may be related, e.g. when it one a reply to an earlier email. The conversationId is the correlationId for all emails that belong to the same "send-reply-reply" thread.** **If no conversationId is supplied, then the system wil be unable to relate this email with any other in the system.**
callback	**string**	**The name of the method to be invoked by the CRM when the untracking process is completed (success or failure)**
cookie	**string**	**The cookie is passed to the callback method to allow the Helper to retrieve contextual information about the email being untracked.**

5.2.3 Return Value

Sometimes the CRM may need to perform some time-consuming operation before it is removed from the database, e.g. to convert the sender and recipient email addresses into internal codes. For this reason, **T42.CRM.UntrackEmail** uses an explicit callback method to notify the Helper when the tracking process is complete, or if it has been cancelled.

The immediate return method from the function indicates only that the CRM has received the request from the Helper; it is not a confirmation that the email has been untracked. This confirmation only arrives by way of the callback method.

If the CRM were to complete the untracking process synchronously, it must still invoke the callback method. Depending on the threading model of the Helper, this may mean that the callback is invoked before the **T42.CRM.UntrackEmail** returns any result, and the Helper must be coded with this possibility in mind.

5.2.3.1 Callback method

The callback method includes the following parameters:

Name	Type	Description

emailIds	**T42Id[]**	**The identifier of the email that the Helper asked to be untracked by the CRM. This is the same value that was passed in the id field of the T42Email parameter to the T42.CRM.UntrackEmail method. The Helper may choose to use this as a correlation value in addition to or instead of the cookie.**
conversationIds	**T42Id[]?**	**A sequence of emails may be related, e.g. when it one a reply to an earlier email. The conversationId is the correlationId for all emails that belong to the same "send-reply-reply" thread.** **If no conversationId is supplied, then the system wil be unable to relate this email with any other in the system.**
success	**bool**	**True if the email was untracked by the CRM. Otherwise an error occurred or the user cancelled the operation.**
errorMessage	**String?**	**An optional error message describing the reason why the email was not untracked. This is used for logging and, depending on the Helper, it may be displayed to the user.** **A typical reason might be "email not found", or "email is locked and can no longer be untracked".**
cookie	**string**	**The cookie value that was passed to the T42.CRM.UntrackEmail method by the Helper.**

5.3 T42.CRM.GetAttachment

This method is used by the CRM to get the contents of an attachment to an email from the Helper. The attachment is pass to the CRM in one or more callback invocations, depending on the length of the attachment and some configurable "maximum length" that may be set in the Helper's configuration. The CRM must be able to rebuild the complete attachment from the pieces ("chunks") that may be passed by the Helper.

If the attachment can fit in a single chunk, then the CRM will only receive a single callback with the entire attachment. At the time of writing this document, the maximum encoded length of a single "chunk" is 4,000,000 bytes, so any attchment that is smaller than this when encoded will be sent in a single callback.

5.3.1 Glue Entity Tag

T42.CRM.GetAttachment does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

5.3.2 Parameters

Name	Type	Description
emailIds	**T42Id[]**	**The email containing the attachment.**
attachmentIds	**T42Id[]**	**Identifies the attachment to be retrieved.**
callback	**string**	**The name of the method to be invoked by the Helper when passing the contents of the attachment to the CRM.**
cookie	**string**	**The cookie is passed to the callback method to allow the CRM to retrieve contextual information about the attachment being passed.**

5.3.3 Return Value

The immediate return method from the function indicates only that the Helper has received the request from the CRM; it does not include the contents of the requested attachment. The contents are sent by the Helper to the CRM only by way of the callback method.

Depending on the threading model of the Helper, the callback method may be invoked before the **T42.CRM.GetAttachment** returns any result, and the CRM must be coded with this possibility in mind, i.e. it must be ready to reeive the attachment data as soon as it invokes **T42.CRM.GetAttachment**.

5.3.3.1 Callback method

The callback method includes the following parameters:

Name	Type	Description
emailIds	**T42Id[]**	**The identifier of the email containing the attachment. This is the same value that was passed to the T42.CRM.GetAttachment method, even if an error occurred. The CRM may choose to use this as a correlation value in addition to or instead of the cookie.**
attachmentIds	**T42Id[]**	**The identifier of the attachment. This is the same value that was passed to the T42.CRM.GetAttachment method, even if an error occurred. The CRM may choose to use this as a correlation value in addition to or instead of the cookie.**
data	**string**	**The contains of the chunk of data encoded as BASE64.**

totalLength	**int**	**The total length of the attachment, as encoded in BASE64. This is used in conjunction with the offset and length parameters to reconstruct the entire attachment. This value is passed for each callback and does not change value.**
offset	**int**	**The offset from the start of the attachment for this chunk. This is always set to zro for the first chunk and incremented by length for each subsequent chunk, so long as success is true.** **Chunks are always sent in order of increasing offset values, so that the CRM may simply concatenate the chunks to rebuild the full attachment.**
length	**int**	**The number of bytes in this chunk. This is set to the minimum of "remaining length" and the "maximum chunk size" as chosen by the Helper. The "remaining length" is equal to the totalLength minus the current offset.**
more	**bool**	**True if there are more chunks to be sent, i.e. offset + length < totalLength. Otherwise, this is the final chunk of data**
success	**bool**	**True if the callback contains a chunk of data from the attachment. False if an error occurred and no more data will be transmitted, whereupon the CRM should discard any chunks of data received so far for this attachment.**
errorMessage	**string?**	**An optional error message describing the reason why the attachment could not be transmited (either completely or in part).** **A typical reason might be "email not found", or "attachment not found"**
cookie	**string**	**The cookie value that was passed to the T42.CRM.GetAttachment method by the Helper.**

5.4 T42.CRM.ShowEmail

This method is used by the CRM to ask the Helper to display an email using whatever mechanism is appropriate for the Helper, with an optional qualifying request from the CRM, where appropriate.

5.4.1 Glue Entity Tag

T42.CRM.ShowEmail does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

5.4.2 Parameters

Name	Type	Description
emailIds	**T42Id[]**	**The email to be displayed.**

5.4.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

5.5 T42.CRM.ShowConversation

This method is used by the Helper to ask the CRM to display a conversation.

5.5.1 Glue Entity Tag

T42.CRM.ShowConversation does not have a predefined Glue entity tag. It is not automatically added as a command-option to a context-menu of methods to invoke.

5.5.2 Parameters

Name	Type	Description
conversationIds	**T42Id[]**	**The CRM should display the conversation with this id. It is expected that the CRM Id can be found within the list of Ids held in this parameter.**
emailIds	**T42Id[]?**	**An optional hint to highlight this email in the conversation.**

5.5.3 Return Value

The Helper returns success or failure (with an optional error message) via the standard GLUE method result values.

6 Tracking Parameter Types

This section is a continuation of the above, but describing the extra parameters that apply only to the methods involved in Tracking emails.

6.1 T42Email

This composite type describes a single email. The data stored in the object includes the sender, subject, recipients, body and optional extra values that are chosen by the Helper.

6.1.1 Members

Name	Type	Description
ids	**T42Id[]**	**Identifiers for the email; globally unique to the system.**
sender	**T42Contact**	**The originator of the email. This might be only the email address, if the T42Contact has not yet been resolved.**
to	**T42Contact[]?**	**The primary recipient(s) of the email.**
cc	**T42Contact[]?**	**The co-recipient(s) of the email. If this parameter is missing then there are no co-recipients.**
bcc	**T42Contact[]?**	**The blind co-recipient(s) of the email. If this parameter is missing then there are no blind co-recipients.**
date	**DateTime**	**The time at which the email was sent, using local time for the sender. This is the *orig-date* field referenced in RFC-2822.**
attachments	**T42Attachment[]?**	**An array of identifiers for the attachments to the email.**
subject	**String**	**The subject field of the email.**
body	**String**	**The body text of the email, formatted as a plain string.**
bodyHtml	**String**	**The body of the email, formatted as an HTML string. If the email was not sent in HTML format, then the Helper must convert the plain-text body into a valid HTML string by wrapping it in the standard tags.**

6.2 T42Attachment

1 This composite type describes a file attachment to an email.

6.2.1 Members

Name	Type	Description
ids	**T42Id[]**	**Identifiers for the attachment, which is unique within a single email.**
emailIds	**T42Id[]?**	**Optional id of the email owning the attachment. This field is only required when a T42Attachment value is used outside the context of the email that owns it.**
name	**String?**	**Display name for the attachment; typically a filename without a path. Therefore, neither the Helper nor the CRM should rely on identifying the attachment using this field.**
sizeHint	**Int?**	**An estimate of the size of the attachment. This is an estimate of the size that the attachment would have if it were saved to disk. It is not the size of the encoded string (e.g. MIME) and it is an advisory-value only, to help with storage optimisations.** **Neither the Helper nor the CRM should rely on this field being present and 100% accurate in all cases (especially when the attachment may be encoded or encrypted In some way).**