

SLG

BizTalk Toolkit Requirements

Prepared by
Mark Tolley

Document Details

File: 1/1

Issue Date: 07 March 2007

ciber

Copyright Notice

©This document and its contents are the property of CIBER Europe Limited and should not be reproduced or discussed outside of CIBER UK Ltd or SLG without the prior permission of CIBER Europe Limited.

CIBER[®], the CIBER logo and Practicology[™] are trademarks of CIBER Inc. and its subsidiaries. Tradenames and trademarks of other companies are the property of their respective owners

Protection of Confidential Information

The information contained in this document describes confidential information concerning SLG's business plans and CIBER's pricing, services and methodology. Development and delivery of this proposal constitutes acknowledgment by both parties that this proposal is proprietary to CIBER and SLG. Both parties will protect and not disclose the contents herein (and any copies thereof), including said confidential information, using the same procedures and requirements by which they protect their own proprietary and confidential information. Both parties will not, in whole or in part, disclose any of the proprietary or confidential information to any person, firm, corporation, association or other entity for any reasons or purpose whatsoever. Nor shall either party make use of any proprietary or confidential information for its own purposes or benefit without the prior express written consent of the other. Neither party will knowingly make this proposal or the information contained herein available, in whole or in part, to current or potential competitors.

Contacts

Any correspondence on this project should be addressed to Les Wright of CIBER UK.

Tel: +44 (0) 20 7355 1101

Mobile: +44 (0) 7973 853430

Email: les.wright@ciber.com

The information contained in this document is subject to change without notice and should not be construed as a commitment by CIBER UK Ltd. It is designed solely to demonstrate CIBER UK Ltd' approach to the design, implementation and support of high-quality structured systems and solutions, and to act as a reference point for products and services approved for supply by CIBER UK Ltd.

Basis of Submission

Acceptance of this proposal or submission shall be subject to agreement of a contract satisfactory to both parties. Any software described in this document is furnished under a licence and may be used or copied only in accordance with the terms of such licence. No responsibility is assumed for the use or reliability of software or equipment that is not supplied by CIBER UK Ltd. This document is valid for 30 days from date of issue.

Document Management

Contributors

Name	Position	Company
Mark Tolley	Lead Architect	CIBER UK

Version History

Version	Date	Updated by	Notes
0.1	12/12/06	N Wood	Document Created
0.2		Mark Tolley	First draft
1.0	13/03/07	Chris Key	Reviewed

References

Ref	File	Notes
A	RBC RC TRA PRG REQ 001 I10 SLG env serv integration reuquirements.doc	Rotherham integration requirements document
B	HLOWNER_HOT_COMPLAINT_TYPES.xls	Newham list of environmental services complaints
C		Schema descriptions for entities

Table of Contents

1	Introduction	1
1.1	Document Purpose	1
2	Vision	2
2.1	Project Background	2
2.2	Project Scope	2
2.2.1	Services to be supported	2
2.2.2	Applications to be supported	3
2.2.3	Pilot Projects	4
2.3	Project Objectives	4
2.3.1	Problem Statement	4
2.3.2	Solution Positioning Statement	4
3	Domain Model	4
3.1	Entities	4
3.2	Other Terms	7
4	Stakeholder and User Descriptions	8
4.1	SLG	8
4.2	Customer	8
4.3	Operator	8
4.4	Department Managers	9
4.5	Back Office Staff	9
4.6	IT Services Managers	9
4.7	IT Services Administrators	9
4.8	IT Services Developers	10
4.9	Suppliers	10
5	Solution Overview	10
5.1.1	Solution Perspective	10
6	Key Features	12
6.1	General Features	12
6.2	Workflow Features	12
6.3	Messaging Features	13
6.4	User Interface Features	15
7	Non-Functional Requirements	17
7.1	Branding / Graphical Requirements	17
7.2	Applicable Standards	17
7.3	System Requirements	17
7.4	Performance Requirements	17
7.5	Environmental Requirements	18
7.6	Documentation Requirements	18
7.6.1	IT Services Administrators	18
7.6.2	Installation Guides, Configuration, and Read Me File	18
7.6.3	Developer Documentation	18
8	Specific Software Requirements	20
8.1	Introduction	20
8.2	Actors	20
8.3	Packages	20
8.4	Package 01: Service Request Handling	21
8.4.1	UC0101 Create Service Request	21
8.4.2	UC0102 Update Service Request	24
8.4.3	UC0103 Query Service Request	27
8.4.4	UC0104 "Push" Service Request Update	29
	Package 02:	33
	Application Administration and Reporting	33
8.5	33	
8.5.1	UC0201 Maintain Application Instances	34
8.5.2	UC0202 Maintain Entities	35
8.5.3	UC0203 Maintain Cross References	36
8.5.4	UC0204 Maintain Subscriptions	38

8.5.5	UC0205 Import Cross References	39
8.5.6	UC0206 View Service Requests Per Hour	40
8.5.7	UC0207 View Service Request Progress	41
9	Business Rules	41
9.1	Business Rule 01: Cross Reference Creation	41
10	Roles and Security	43
10.1	Roles	43
10.2	Security	43
11	Data Requirements	44
11.1	Existing Data Definitions	44
11.2	Data Conversion	44
12	Software Architecture Overview	45
12.1	Logical View	45
12.2	Deployment View	46
13	Risks & Constraints	48
13.1	Technical Risks	48
13.2	Schedule Risks	48
13.3	Constraints	48
14	Training Needs	49
15	Project Approach	50
15.1	Project Management	50
15.2	Time Boxing	50
16	Estimates	51
16.1	Assumptions	51
16.2	Estimation	51
17	Project Schedule	53
17.1	Milestones	53
17.2	Project Plan	53
18	Glossary	54
Appendices		Error! Bookmark not defined.
A	Insert first appendix title	Error! Bookmark not defined.

1 Introduction

1.1 Document Purpose

This document is the result of the Discover stage of the CIBER custom development process. In this stage, we identified the business objectives for the application, and some of the issues and concerns surrounding the development process. Here, we identify the scope and define the business objectives of this project in sufficient detail to enable a thorough and accurate application design.

Note that this document describes the general requirements for the toolkit only. Detailed requirements for the pilot projects at Rotherham and Newham will be documented separately (see reference A for the requirements for Rotherham).

2 Vision

The purpose of this section is to collect, analyze, and define high-level needs and features of the BizTalk toolkit. It focuses on the capabilities needed by the stakeholders, and the target users, and why these needs exist. The details of how the BizTalk toolkit fulfils these needs are detailed in the use-case specification and supplementary specifications.

2.1 Project Background

The SLG (Shared Learning group) provides a forum through which a number of local authorities are able to share ideas and best practice solutions. The current members are Newham Borough Council, Rotherham Metropolitan Borough Council, Wakefield Metropolitan District Council and Sunderland City Council.

All these councils have implemented a CRM-based approach to receiving requests for services from members of the public and from businesses. Citizens can specify their requirements through a variety of channels including telephone, letter, the internet and face-to-face. Details of the contact are recorded in CRM and the service request is delivered to the council department responsible for implementing them. This allows service requests to be presented in a consistent fashion to departments regardless of the channel through which they arise and enables a dedicated contact centre to relieve the pressure on back office staff. Some CRM applications also provide the capability for centralised KPI monitoring across departments, although this functionality is not implemented by all councils.

At present, service requests taken through the CRM system are typically passed to departments via paper, email or phone call. Back office staff then enter the requirements as a job or task in the back office application used by their department. This process is inevitably slow and prone to error. A far better solution would be for service requests recorded in CRM to be passed to back office systems automatically.

Back office applications are typically stand-alone and APIs for integration with other systems, if any, are often limited and poorly documented. Integration with CRM therefore can be a considerable challenge. This can be alleviated by use of a middleware infrastructure, and all councils in the SLG have agreed to use Microsoft BizTalk Server 2006 for their integration requirements.

In addition to the problems of the different transport methods, objects and request-response patterns used by individual application APIs, passing of service requests between applications involves a number of common issues, including:

- De-duplication of multiple requests for the same service (eg several people report the same abandoned car)
- Maintenance of cross-references between different systems for requests, people and locations
- Mapping between the business entities understood by the different back office applications
- Handling of delivery and data errors
- Auditing
- Passing of acknowledgments back to source applications
- Queries for service request status
- Notification of change in service request status (eg from open to complete or cancelled)

These are best addressed by using a customisable framework that is built on the integration middleware, and it is this that the toolkit aims to provide.

2.2 Project Scope

2.2.1 Services to be supported

Local authorities provide a wide range of services and supporting all of them in the first iteration of the toolkit would be prohibitively complex. It has therefore been agreed that the toolkit will initially only support “street scene” services provided by environmental services departments. These have been selected because they are high volume but relatively uncomplicated - it is not necessary for citizens to provide credentials to request services and the required information is generally restricted to the

name and contact details of the requesting citizen or company, the location where the service is required and a few particulars. Street scene services include:

- Missed bin collections (domestic or commercial)
- Collections of bulky waste such as old settees or fridges
- Street cleaning (littering, graffiti, dead animals, dog fouling, fly tipping)
- Removal of abandoned cars
- Collection of stray dogs
- Provision of recycling containers

See references A and B for the specific service types used by Rotherham and Newham.

It has been agreed that although the toolkit will focus on these services, it should be designed in such a way that other services can be added at a later date. A full list of the services provided by UK local authorities can be found at <http://www.esd.org.uk/standards/lgsi/viewer/viewer.aspx>.

2.2.2 Applications to be supported

All the councils are using different CRM and back office applications. The following applications have been identified as candidates for integration:

Application	Council	Notes
Siebel CRM	Rotherham	Fully featured Service request details are stored as "smart script" question and answer sessions.
LA CRM	Newham	An open source CRM system that is customised for local authorities. Supports calls only, no cases.
SAP CRM	Sunderland	
Authority Public Protection	Rotherham	Supplied by Capita. Formerly known as "Flare". Supports all environmental services.
Mayrise	Newham	Newham's environmental services application. Has an API, but documentation is not publically available.
MBM	Sunderland	Supplied by Northgate. General environmental services system.
[Custom SQL/Access application]	Rotherham	In-house application used for missed bins.
CAPS UNI-Form	Newham, Rotherham	General local authority application supporting environmental health and licensing services <i>inter alia</i> .
Email	Rotherham	Some services have no supporting back end application – service requests are delivered to a particular departmental inbox.

Note: It is assumed that for an application to be supported by the toolkit, some sort of API exists or can be written that supports the following on-demand functionality:

Mandatory

- Create service request for each service type supported by the application (return cross-references highly desirable but only mandatory if service request updates are supported)

Optional

- Update service request (return update ID highly desirable but not mandatory)
- Query service request – return service request details and status
- Query customer – given customer name / address, return customer ID referenced in service request
- Query address – given house name/number and post code, return ID of address referenced in service request

- “Push” update: when a service request has an action taken on it.

If no API exists or can be written for an application, “integration” will consist of automated emails to the application administrator for the back-end system.

2.2.3 Pilot Projects

The initial implementation of the toolkit will be for Rotherham and will integrate Siebel CRM with Authority Public Protection. The toolkit will subsequently be implemented at Newham for LA CRM and Mayrise.

2.3 Project Objectives

The toolkit will identify the best practice approach for local authority enterprise integration of service requests. The toolkit will provide a flexible framework which can be easily customised, deployed and reused by various local authorities.

2.3.1 Problem Statement

The Project Problem Statement that explains *WHY* the project has been initiated and the perceived successful solution:

The problem of...	Service requests collected from citizens through the contact centre, web and other channels and entered in CRM must be manually entered in departmental back office systems
Affects...	Customers, contact centre operators, back office staff, council managers.
The impact of which is...	Errors and slow response to customers; unhappy staff; KPIs cannot be adequately monitored; services are not standardised.
A successful solution would...	Reduce workload on back office staff, reduce errors, improve the speed and efficiency of service provision to customers, allow standardisation of service delivery, allow monitoring of KPIs.

2.3.2 Solution Positioning Statement

The Solution Positioning Statement describes *WHAT* solution is being developed:

For...	council IT departments
Who...	implement integrations between CRM and back office applications using BizTalk Server 2006
The solution is...	A best practice local government integration toolkit
That...	Provides documentation and core code through which local authority IT departments are able to implement a messaging hub allowing the synchronous and asynchronous transportation and transformation of service requests between a wide variety of systems
Unlike the...	current manual processes
This solution is...	easy to manage and upgrade, allowing local councils to administer and extend the integration framework.

3 Domain Model

A “domain model” is an abstract conceptual model of a system defining the entities that make it up and the relationships between them. It defines a common vocabulary that can be used when developers and stakeholders are discussing the project.

3.1 Entities

The following entities have been defined for the domain model for this project:

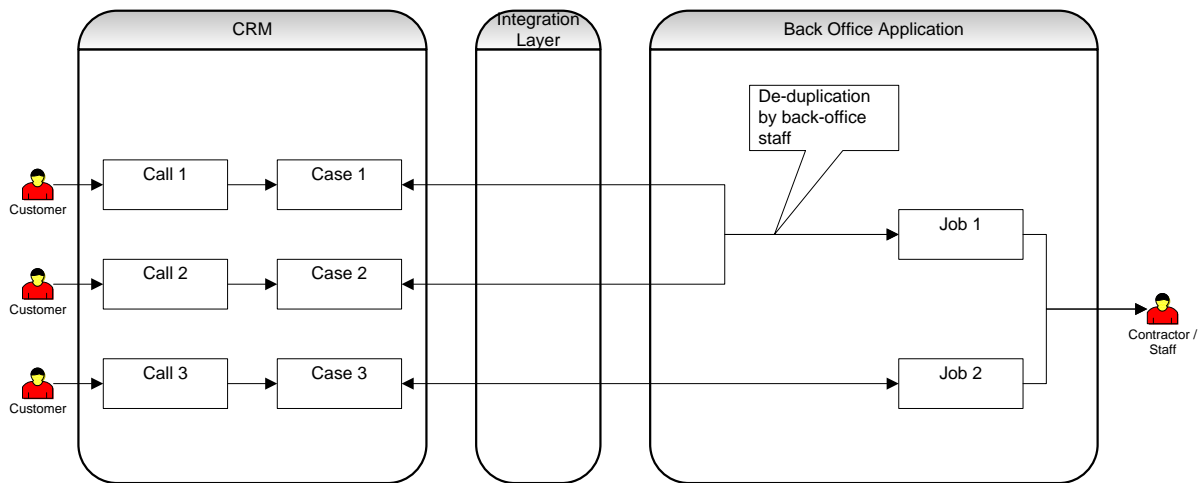
Entity	Aliases	Description
--------	---------	-------------

Application	System	A computer application or system that is supplied by a single supplier and is treated as a separate entity for administration and maintenance purposes. Must have a standard name.
Application Instance		A particular instance or installation of an Application (eg a new version). Used in the BizTalk built-in cross-referencing system. Must have standard name (which can be the same as the Application name).
Call	Contact, Request, Enquiry (Siebel)	A single contact (telephone call, email, letter, fax, text etc) from a customer. A single call could trigger the creation of one or more Cases and a single Case may be referred to by one or more Calls. A call has one associated Customer.
Case	Service Request, Process, Workflow	A requirement for council staff or contractors to perform a service at a particular location in order to fulfil requests from one or more customers. A Case has a status (open, closed or cancelled) and Location and <i>either</i> corresponds to a Call <i>or</i> to a Job, depending on the council service request fulfillment architecture*.
Cross Reference	CrossRef	An entity that associates an Identifier in an Application with an Identifier that is common across Applications.
Customer	Citizen, Business	The organisation or individual who has requested a service from the council. Includes the contact details (email, telephone, fax number etc) for the customer.
Entity		A business entity type (eg Call, Case, Location, Service etc). Has a standard name.
Entity Instance		An instance of a business entity (eg a particular Case). Must have an Identifier attribute (see below).
Identifier	ID	An attribute that uniquely identifies an Entity Instance.
Job	Workflow, Process	A workflow (sequence of tasks) that must be performed to close one or more Cases. A job has a status (eg awaiting staff assignment, in progress, complete).
Message		A document containing data about a single entity instance (usually a Case) that is sent from a source Application to a target Application.
Location	Place, Address	A description of a geographical location. It can be a contact address for a customer or a location where a service is to be performed. Note that a Location can be more precise than an address (eg "In the middle of the field next to number 43, Gold Street").
Operator	Contact Centre Staff	A council staff member who works in the Contact Centre taking calls from Customers.
Service		A service provided by a council, implemented technically as a workflow type of which a Case is a specific instance. E.g. Household waste collection, Abandoned vehicle etc. NB Does not necessarily

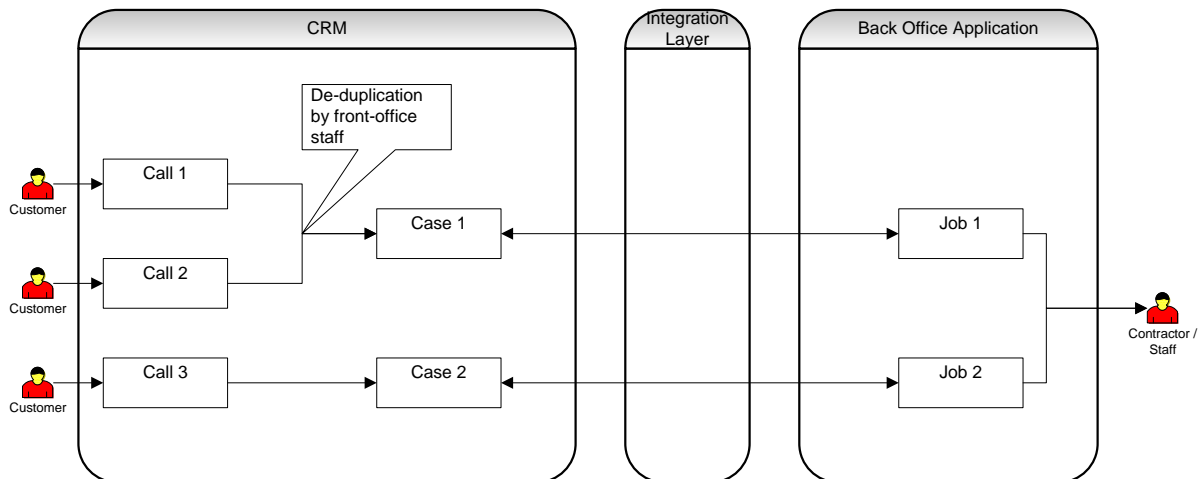
		correspond to a service definition in the LGSL.
Subscription		A record that an Application Instance will publish and/or receive Service Requests for a particular Service/SubService.
SubService	Subject	An optional sub-classification of a Service that is used by some CRM and back office applications. Eg missed bin complaint, green box request etc.

* Because of the differing capabilities of CRM applications and back office applications, the definition of a “Case” is intentionally ambiguous. There are two possible service request fulfillment architectures that the toolkit will support:

1: Case de-duplication in back office



2: Case de-duplication in front office



In architecture 1, it is the responsibility of the back office application to maintain the Case-to-Job associations (which are many-to-one) and issue a “push” update for each associated Case when a Job status is updated. The Call-to-Case association is one-to-one.

In architecture 2, the Case-to-Job association is one-to-one and can be maintained by the integration layer. Call-to-Case association is many-to-one and would be maintained by the CRM application. The CRM application would be responsible for issuing “Case complete” notifications to customers.

See reference C for the current list of entity attributes.

3.2 Other Terms

The following table defines other terms that are used in this document.

Entity	Aliases	Description
Ack List		A list of Source Application Instances to which a Target Application Instance is expected to return an Acknowledgment when creating or updating a Service Request. Also used to identify the Target Application that should respond to a Query from the Source Application Instance.
Ack Message		An internal representation of an Acknowledgment from a Target Application Instance (including Case details returned from a query) within BizTalk
Common Identifier		An Identifier for an entity instance that is common across all Application Instances in a council.
CRM Ack Message		A Message that is sent by BizTalk to CRM to acknowledge receipt of a new service request or service request update.
CRM Case Creation/Update Message		A message that is sent by CRM to BizTalk when a Case is created/updated.
CRM Query Message		A Message that is sent by CRM to BizTalk to retrieve details for a Case.
CRM Case Details Message		A message sent by BizTalk to CRM containing the details of a single Case.
Global Identifier		An Identifier for an entity instance that is globally unique across councils and other organisations (eg UPRN for property)
Local Identifier		An identifier for an entity instance in an Application Instance
Push Update		A Case update (usually a change in status or a new annotation) that is “pushed” from a back office Application Instance to CRM.
Service Request Message		An internal representation of a service request / case (creation, update or query) within BizTalk
Source Application Instance		The Application Instance that is the source for a Message.
Standard Name		The standard name used for an Application, Application Instance or Entity in the cross-referencing system (eg Authority, Siebel, Case, Call, Service etc). Used in maps that do cross-referencing. NB Standard names are case-sensitive – “Case” is not the same as “case”.

		A standard name must be unique.
Job Creation /Update Message		A message sent by BizTalk to a Target App to request creation of a new Case or Job.
Job Creation/Update Ack Message	Response, Acknowledgment	An Acknowledgement received from a Target Application Instance that a Service Request has been created or updated.
Query Message		A query sent by BizTalk to a Target Application Instance to retrieve details of a Case.
Job Details Message		The response to a Query from a Target Application Instance containing status details for a single Job/Case.
Target Application Instance		The Application Instance that is the target for a Message.

See Appendix A for a glossary of other terms and acronyms used.

4 Stakeholder and User Descriptions

This section describes the stakeholders (groups with an interest in the project who are not necessarily users) and users (people who will be using the solution on a regular basis). It does not describe specific requirement requests but rather background and justification for why the requirements are needed.

4.1 SLG

Representative	Graeme Hutchinson
Description	Shared Learning Group
Type	Project Sponsors
Responsibilities	Agree scope and generic requirements, oversee development
Success Criteria	Project delivered to time and budget
Involvement	Analysis and prioritisation of business needs. Control of costs. Approve Requirements and Design documents.
Deliverables	Analysis document, design documents, code, other documentation

4.2 Customer

Representative	Varies
Description	Council customer (individual or organisation) for whom services are performed.
Type	Non-technical User
Responsibilities	Submit Calls (new service requests, updates, cancellations) through self-service interfaces or via an Operator.
Success Criteria	Service provided as requested
Involvement	
Deliverables	

4.3 Operator

Representative	Varies
Description	Contact centre staff
Type	Non-technical User
Responsibilities	Submit and update Cases in CRM on behalf of customers.

	Query Case status on behalf of customers.
Success Criteria	Case information available and up to date. No additional work is required in entering, submitting or querying a Case.
Involvement	
Deliverables	

4.4 Department Managers

Representative	Varies
Description	Heads of council departments
Type	Non-technical Stakeholder
Responsibilities	Report to councillors and central government on service provision and budgets
Success Criteria	Improvement in service response times Reduced costs
Involvement	
Deliverables	

4.5 Back Office Staff

Representative	Varies
Description	Departmental staff and contractors responsible for delivery of services
Type	Non-technical User
Responsibilities	Update Cases in back office Applications
Success Criteria	Cases appear as jobs or tasks in back office Application without manual intervention in a timely fashion. Information in Cases is correctly entered and cross-referenced for all Service types. Updates to Cases appear in CRM application without manual intervention.
Involvement	
Deliverables	

4.6 IT Services Managers

Representative	Varies
Description	Managers for development projects and administrative staff within the council
Type	Technical Stakeholder
Responsibilities	Determine IT implementation strategies Determine quality of service standards for Applications Manage relationships with Application Suppliers Project management for development projects Management of IT services administrative staff
Success Criteria	New solution does not disrupt existing Applications New solution does not have large maintenance overheads
Involvement	Determine software security, UI and quality standards
Deliverables	Design document (including physical architecture)

4.7 IT Services Administrators

Representative	Varies
Description	Council staff responsible for installation and maintenance of council IT systems
Type	Technical Stakeholder
Responsibilities	Install and configure the solution. Update configuration when new Applications and supported Services are added. Receive and act on alerts when Applications go offline Diagnose and respond to technical errors (eg messages not delivered due to

	missing data)
Success Criteria	Solution is easy to deploy and maintain Application status is readily available Appropriate alerts are received when a message fails to be delivered Message delivery failures are easy to diagnose and remedy
Involvement	Determine requirements for error handling, alerting, configuration and reporting.
Deliverables	Design document (including physical architecture), deployment document, systems operation documentation, reporting and administration UIs.

4.8 IT Services Developers

Representative	Varies
Description	Council staff responsible for developing integration solutions with Applications.
Type	Technical Stakeholder
Responsibilities	Development of new messages and interfaces. Update existing Application interfaces in response to changes in business requirements.
Success Criteria	Easy to develop interfaces for new Applications. Easy to update existing interfaces.
Involvement	Develop APIs to back office Applications (where this can be done in-house)
Deliverables	Analysis document, design document, source code

4.9 Suppliers

Representative	Varies
Description	Third party suppliers of back office Applications
Type	Technical Stakeholder
Responsibilities	Development of Application APIs
Success Criteria	Minimal custom development of API
Involvement	Develop APIs to back office Applications
Deliverables	Interface requirements document

5 Solution Overview

This section provides a high level view of the capabilities, interfaces to the external systems, and the system configuration.

5.1.1 Solution Perspective

Figure 1 shows a high level overview of the way in which the solution will fit within the existing council architecture.

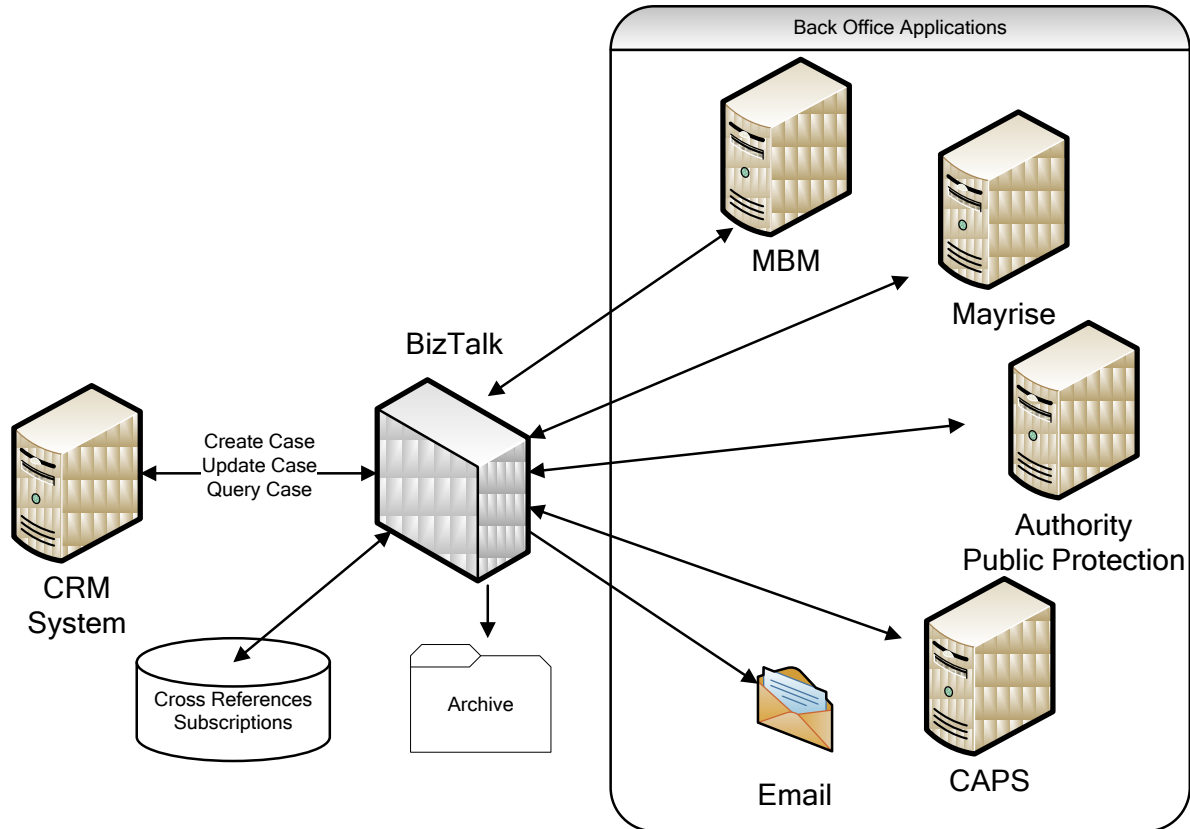


Figure 1 - System Context Diagram

The toolkit will define an architecture for the BizTalk components and provide standard facilities for mapping, cross-referencing, acknowledgments, configuration of message routing, error handling and auditing. In addition, guidance and templates for development of specific application interfaces will be provided.

6 Key Features

This section offers a concise description of solution functions in outline form. Functions that are to be excluded are marked as such.

6.1 General Features

ID	Feature	Reason or Benefit	Stakeholder/ User Priority
1.	The Solution should be generic and usable at as many councils as possible	Key Feature	Must
2.	“Out of the box” functionality vs custom application-specific configuration/development should be clearly stated	Ease of deployment	Must
3.	Solution should use generally accepted standards such as BS7666 and BS8766.	Usability across multiple councils	Must
4.	The CRM and back-end application functionality requirements needed to support the solution should be clearly stated.	Ease of deployment	Must

6.2 Workflow Features

ID	Feature	Reason or Benefit	Stakeholder/ User Priority
5.	The solution must support Case creation and management <i>either</i> in CRM <i>or</i> in the back-end application.	Compatibility with existing business processes	Must
6.	<p>The solution must enable application integration for the following services:</p> <ul style="list-style-type: none"> • Street Litter Enforcement • Fly Tipping Enforcement • Dog Fouling Enforcement • Stray Dogs • Missed Household Collection • Household Waste Complaint • Missed Commercial Collection • Commercial Waste Complaint • Recycling Sites • Customer Requesting New Domestic Recycling Container • Customer Requesting Information About Assisted Collection • Bulky Items 	Support for Rotherham pilot and anticipated Newham implementation	Must

	<ul style="list-style-type: none"> Abandoned Vehicles Pest Control 		
7.	The solution should be designed so that other services can be added at a later date.	Future development	Must
8.	<p>The solution must support application integration for the following general processes:</p> <ul style="list-style-type: none"> Send new Case from front office to back office Send update of existing Case from front office to back office Send update of existing Case from back office to front office Retrieve details of existing Case from back office and provide to front office (i.e. query Case) 	Maintenance of Case state	Must
9.	<p>As a minimum the solution should support the following unsolicited updates from the back office to the front office:</p> <ul style="list-style-type: none"> Acknowledgment of Case creation First response update Case Closed update 	Maintenance of Case state	Must
10.	The Solution must support multiple back office Applications for different service types, eg pests and dogs may go to one back office Application, street scene to another.	Support for existing Applications	Must
11.	The Solution must support delivery of a single Case to more than one back office Application (eg environmental services back office Application plus auditing Application)	Support for existing business processes	Must
12.	The solution must support grouping of Cases (eg several reports of the same abandoned car) <i>either</i> by contact centre staff in CRM <i>or</i> in the back-end application by environmental services staff.	Prevention of duplicate Jobs	TBD
13.	The solution should include a standard approach to system error handling (message data errors, application interfaces not available)	Maintainability	Must
14.	The solution should enable a new Case to be added to an existing Job in a back office Application by allowing a Job ID to be specified in the Service Request Message by the CRM Application.	Prevention of duplicate Jobs	Must

6.3 Messaging Features

ID	Feature	Reason or Benefit	Stakeholder/ User Priority
----	---------	-------------------	-------------------------------

15.	The solution must support messages in XML and flat file formats.	Generic support for back end applications	Must
16.	The solution should allow for Message types other than single Cases (eg Case List for queries, appointment queries and responses)	Future development	Should
17.	The solution must use a standard common data model for business entities. It must support “side by side” versioning so that schemas can be updated from time to time as business requirements change without affecting existing integrations.	Standardisation of business processes.	Must
18.	The solution must use a standard approach to transform application-specific messages to “canonical” messages and vice versa, applying specific transformation rules where necessary.	Standardisation of mapping technology	Must
19.	<p>The solution should provide functionality to manage creation and maintenance of Cross References during message mapping for the following:</p> <ul style="list-style-type: none"> • Cases • Services / SubServices <p>The solution should be capable of the following:</p> <ul style="list-style-type: none"> • Identify Common Identifier for entity from the Application Instance Local Identifier • Identify Local Identifier for entity from Common Identifier (NB mapping between a local identifier and a common identifier must be 1:1 ie no many-to-one relationships). • For an Application Instance entity, create a new Common Identifier and create Cross References to Application Instance Identifier for both Source and Target Application Instances. 	Standardisation of mapping technology	Must
20.	The cross referencing system should allow additional entities (eg Locations and Customers) to be added.	Generic support for business processes	Should
21.	The solution should fulfil the needs of both synchronous and asynchronous application scenarios (ie synchronous and asynchronous transport mechanisms).	Generic support for back end applications	Must
22.	<p>The solution must support the following transport methods to/from back office Applications:</p> <ul style="list-style-type: none"> • Web service • Filedrop • Email 	Generic support for back end applications	Must
23.	The solution should allow for the provision of the following information as a minimum for unsolicited	Maintenance of Case state	Must

	<p>update messages from the back office (note that not all of these may be provided by all back office systems):</p> <ul style="list-style-type: none"> • Service Request ID • Status • Target completion date • Last update date • Assigned officer • Summary text 		
24.	<p>The solution should perform basic validation of Messages received from Applications (mandatory data items are present, data items are of correct data type). If a Message fails validation, an error should be logged, and, if the Source Application supports Acknowledgments, an error number, description and source should be returned in the Acknowledgment message.</p>	Early identification of problems.	Must
25.	<p>The solution should be capable of alerting administrators when repeated attempts to deliver a message to a target Application fail, or when the timeout period for reception of a response Message from an Application is exceeded.</p>	Early identification of problems.	Must
26.	<p>The solution should be able to save all received messages to a file store for auditing purposes</p>	Auditing	Must
27.	<p>The common schema for Cases (service requests) should contain the following data item types:</p> <ul style="list-style-type: none"> • Data items common to all Cases (case details, location details, customer details, feedback, payment details) • Named fields for “common” data items for specific service request types (eg make of abandoned vehicle, description of stray dog etc) • AdditionalInfo fields for request details that are specific to a council (eg Rothercard number) 	Ease of mapping, generic support for back end applications	Must

6.4 User Interface Features

ID	Feature	Reason or Benefit	Stakeholder/ User Priority
28.	<p>The solution should include an easy-to-use user interface which will provide administrative functionality to IT services administrators to configure and manage the solution. This should include:</p>	Maintainability	Must

	<ul style="list-style-type: none"> • Cross reference creation and deletion • Addition and configuration of new Application Instances and association with specific Services • Tracking of Messages passing through the integration layer 		
29.	<p>The solution should provide basic reporting. This should include as a minimum:</p> <ul style="list-style-type: none"> • the ability to view aggregate information (e.g. number of Cases of a particular type submitted within a specified timeslot) • the ability to view information about specific Cases and updates (e.g. the history of a Case given a specific ID). 	Identify issues	TBD
30.	The administrative UI should provide a batch import facility allowing Cross References to be imported from a flat file or Excel spreadsheet	Ease of configuration	TBD

7 Non-Functional Requirements

This section describes the non-functional requirements of the solution.

7.1 Branding / Graphical Requirements

None.

7.2 Applicable Standards

Standard Requirement	Reason or Benefit	Stakeholder/ User Priority
Messages conform to XML version 1.1 standards	Current XML standard	Must
Messages conform to XSD version 1.0 standards	Current XSD standard	Must
Messages use Unicode UTF-8 encoding	Standard internal BizTalk encoding	Must
Address data fields should be mappable to/from BS7666	Use of consistent address format	Must
Personal information should be mappable to/from BS8766	Use of consistent personal data format	Must
Services for specific should be mappable to/from the local government services list (LGSL)	Use of consistent service mappings	TBD

7.3 System Requirements

System Requirement	Reason or Benefit	Stakeholder/ User Priority
Active Directory	Use of windows integrated authentication for data access	
Microsoft Windows Server 2003		Minimum Requirement
Microsoft SQL Server 2000 / 2005		Minimum Requirement
Microsoft BizTalk Server 2006 enterprise		Minimum Requirement

7.4 Performance Requirements

Performance issues can include such items as user load factors, bandwidth or communication capacity, throughput, accuracy, and reliability or response times under a variety of loading conditions.

Note: These performance requirements are guesswork. To be replaced with reliable performance requirements.

Performance Requirement	Reason or Benefit	Stakeholder/ User Priority
-------------------------	-------------------	-------------------------------

Support for up to 10 service requests per second	Reasonable concurrency?	TBD
Maximum time for complete message processing (eg CRM submit request to receive ack) of 1 min	Maximum acceptable "wait time" for customer.	TBD
Support average 200 new Cases per day, maximum 500.	Reasonable average?	TBD
Support average 200 Case updates per day, maximum 500.	Reasonable average?	TBD
Support average 1000 "push updates" per day, maximum 5000.	Reasonable average?	TBD
Up time of 99%	Standard for online application	TBD
Reinstallation of application on rebuilt server should take less than 4 hours	Recovery in case of server failure	TBD

7.5 Environmental Requirements

Environmental Requirement	Reason or Benefit	Stakeholder/ User Priority
Error reporting should integrate with MOM or other application log monitoring used by IT Services	Standardised alerting for errors in IT Services	Must
It should be possible to deploy interfaces for new applications without requiring application down time.	Maintenance of up time	Must
It should be possible to deploy new versions of core schemas without disrupting existing processes or recompiling / reinstalling existing code.	Maintenance of up time	Must

7.6 Documentation Requirements

This section describes the documentation that must be developed to support the toolkit.

7.6.1 IT Services Administrators

A user manual for IT services administrators will be provided that contains the following:

- Description of the administration user interface
- Description of the reporting interface
- "How tos" for common administrative operations (install patches, configure new application end point, maintain Cross References, view reports)
- Advice on troubleshooting common problems (lost messages, timeout errors)

7.6.2 Installation Guides, Configuration, and Read Me File

- Minimum server specifications
- Software pre-requisites
- Step-by-step instructions for deploying solution components
- Configuration settings

7.6.3 Developer Documentation

- Requirements document
- Design document
- Component specifications

- Source code
- Code comment report
- Application interface developer's guide
- Application interface specification template
- Test harnesses for core components
- System Test scripts following use cases below

8 Specific Software Requirements

The following sections contain all the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements and testers to test that the system satisfies those requirements.

8.1 Introduction

This section describes the processes and workflows that are supported by the SLG toolkit using 'use cases'. A 'use case' is a description of how end-users (people or applications) will use the delivered software. It describes a task or a series of tasks that users will accomplish using the software, and includes the responses of the software to user actions.

Note: Terms used in the Domain Model (see above) are shown in **bold**.

8.2 Actors

An actor is someone or something outside the solution that interacts with it. Each use case delivers an observable benefit or result to the actor(s) involved.

The following actors initiate use cases:

Actor Name	Description
Customer	An individual who creates or updates a service request from a council through a self-service website, either on their own behalf or for an organisation for which they are an agent.
Operator	A staff member in the council contact centre who creates, updates or queries a service request in the CRM application on behalf of a customer.
Department Administrator	A council staff member who administers a job in a back office application
Department Employee	Council staff or contractor responsible for performing the job.
IT Services Administrator	Council staff member responsible for maintenance and deployment of IT systems, including troubleshooting of errors.
CRM Application	The CRM application in use at the council.
Target Application	A back-office Application Instance.

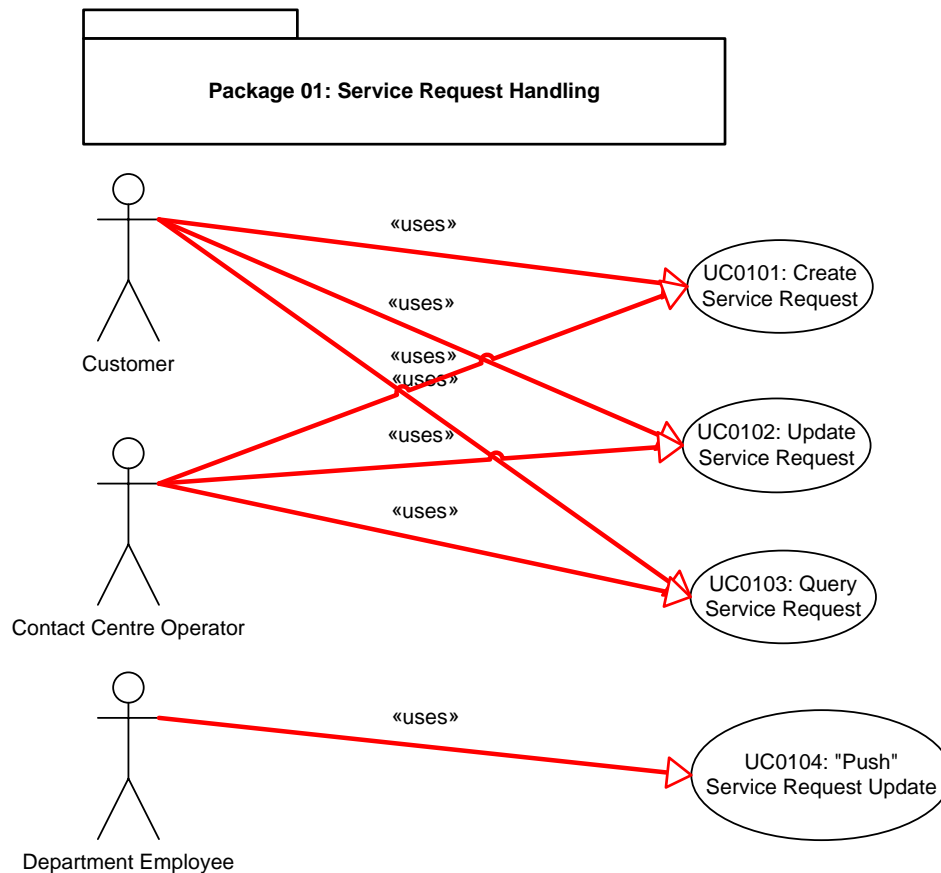
8.3 Packages

The use cases have been divided into the following packages:

Package 01: Service Request Handling

Package 02: Application Administration and Reporting

8.4 Package 01: Service Request Handling



8.4.1 UC0101 Create Service Request

This use case covers the interactions which occur when a Customer or Operator requests a new service from the council.

Goal

Job(s) for new service request created in back office Application Instance(s), acknowledgment returned to CRM.

Start

The use case starts when a case has been created within the CRM system as a result of a telephone, letter, fax, web form, email or third party.

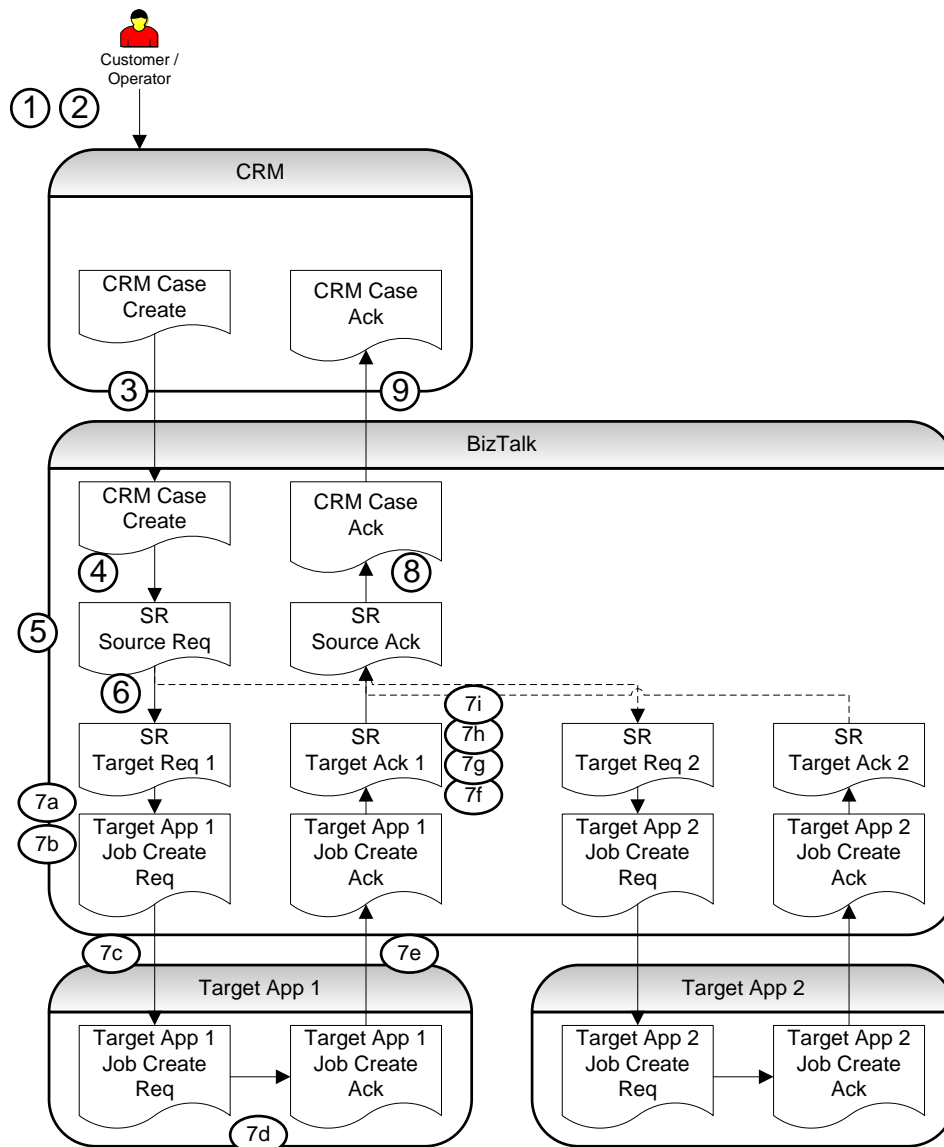
End

The use case ends when the details of the new service request can be viewed in the appropriate back-end Application Instance and a success Acknowledgment has been returned to the CRM Application Instance.

Pre-requisites

- The call made by the Customer contains a request for a service that the council must provide.
- Customer has provided enough information to create the request in the CRM system.
- Payment has been made and authorised for CRM-based payment systems.

Basic Flow



1. The Customer contacts the council by email, phone, letter or fax, or through the self-service website.
2. An Operator (or the CRM interface, if through the self-service website) creates a Call in the CRM system which includes the contact details for the Customer, and a Case containing the details of the request.
3. The CRM API passes the Case details to BizTalk as a CRM Case Creation Message, which creates a Service Request Message with a RequestType of "create" (see ref C for details of mandatory and optional fields).
4. As part of Service Request Message creation, the system retrieves Common IDs that match CRM Local IDs for the Customer, Location, Service and SubService.
5. The system validates the Service Request Message to ensure that all mandatory fields are present and all fields are of the correct data type.
6. BizTalk determines the list of Target Application Instances (other than the Source Application Instance) that have been configured to accept messages of this service type.
7. For each Target Application Instance in turn:
 - a. The Service Request Message is mapped to a Job Creation Message for the Target Application Instance

- b. As part of message creation, the system retrieves **Local IDs** for **Customer**, **Location**, **Service** and **SubService** for the **Target Application Instance**.
 - c. The Job Creation Message is sent to the Target Application Instance API.
 - d. The **Target Application Instance** creates the new **Job**, or associates the Job with the new Case ID if a Job ID has been provided.
 - e. The system receives a **Job Creation Ack Message** from the **Target Application Instance** (see reference C for details of mandatory and optional fields). If the **Target Application Instance** cannot return a **Target App Ack Message** (because it is “fire and forget” eg email), steps 7f to 7i are skipped.
 - f. The Job Creation Ack Message is mapped to a standard Ack Message.
 - g. The system validates the Ack Message to ensure that all mandatory fields are present and are of the correct data type.
 - h. The system takes the **Service Request Message** and the **Ack Message** and creates new Cross References according to **Business Rule 01**.
 - i. If the Source Application Instance is in the Target Application Instance’s Ack List, the JobID value is recorded.
8. The system maps the Service Request Message to a CRM Ack Message, including the AckID.
 9. The system sends the CRM Ack Message to the CRM Application Instance.

Alternate Flow AF1 - Service Request Message fails validation

1. If in BF step 5, the Service Request Message fails validation, the system records an error in the BizTalk application log.
2. The system constructs a CRM Ack Message, specifying the error in the ErrorDescription field.
3. The system returns the CRM Ack Message to the CRM Application Instance.

Alternate Flow AF2 - Timeout error contacting Target Application Instance

1. If in BF step 7c a timeout error is received while trying to contact the Target Application Instance API, the system re-sends the message (in case it was a transient error). If the re-delivery is successful, flow resumes from step 7e.
2. If a timeout error occurs with the re-sent message, the system records an error the BizTalk server application event log and constructs a CRM Ack Message, specifying the error the ErrorDescription field and the Target Application Instance in the Error Source field
3. The system returns the CRM Ack Message to the CRM Application Instance.

Note: The system does *not* attempt to deliver the message to any other Target Application Instances and does *not* attempt to “roll back” previous successful deliveries.

Alternate Flow AF3 - Target Application Instance returns error in Ack Message

1. If in step 7e the Target App Ack Message has a non-empty ErrorDescription, the system records an error in the BizTalk Server application event log.
2. The system constructs a CRM Ack Message, specifying the error the ErrorDescription field and the Target Application Instance in the ErrorSource field
3. The system returns the CRM Ack Message to the CRM Application Instance.

Note: The system does *not* attempt to deliver the message to any other Target Application Instances and does *not* attempt to “roll back” previous successful deliveries.

Scenarios

- Create Abandoned Car request
- Create Vehicle Obstruction request
- Create Street Cleaning (Littering) request
- Create Fly Tipping request
- Create Dog Fouling request

- Create Bulky Waste request
- Create Missed Bin (Domestic) request
- Create Missed Bin (Commercial) request
- Create Stray Dogs request
- Create Dead Animals Removal request
- Create Recycling Bags request
- Create request – Service Request fails validation
- Create request – Target Application Instance unavailable
- Create request – Error from Target Application Instance

8.4.2 UC0102 Update Service Request

This use case covers the interactions which occur when an update to the details of a service request are made in the CRM Application Instance and need to be distributed to back office Application Instances.

Goal

Job(s) for the service request are updated in back office Application Instance(s), acknowledgment returned to CRM.

Start

The use case starts when a Case has been updated within the CRM system as a result of a telephone, letter, fax, web form, email or third party.

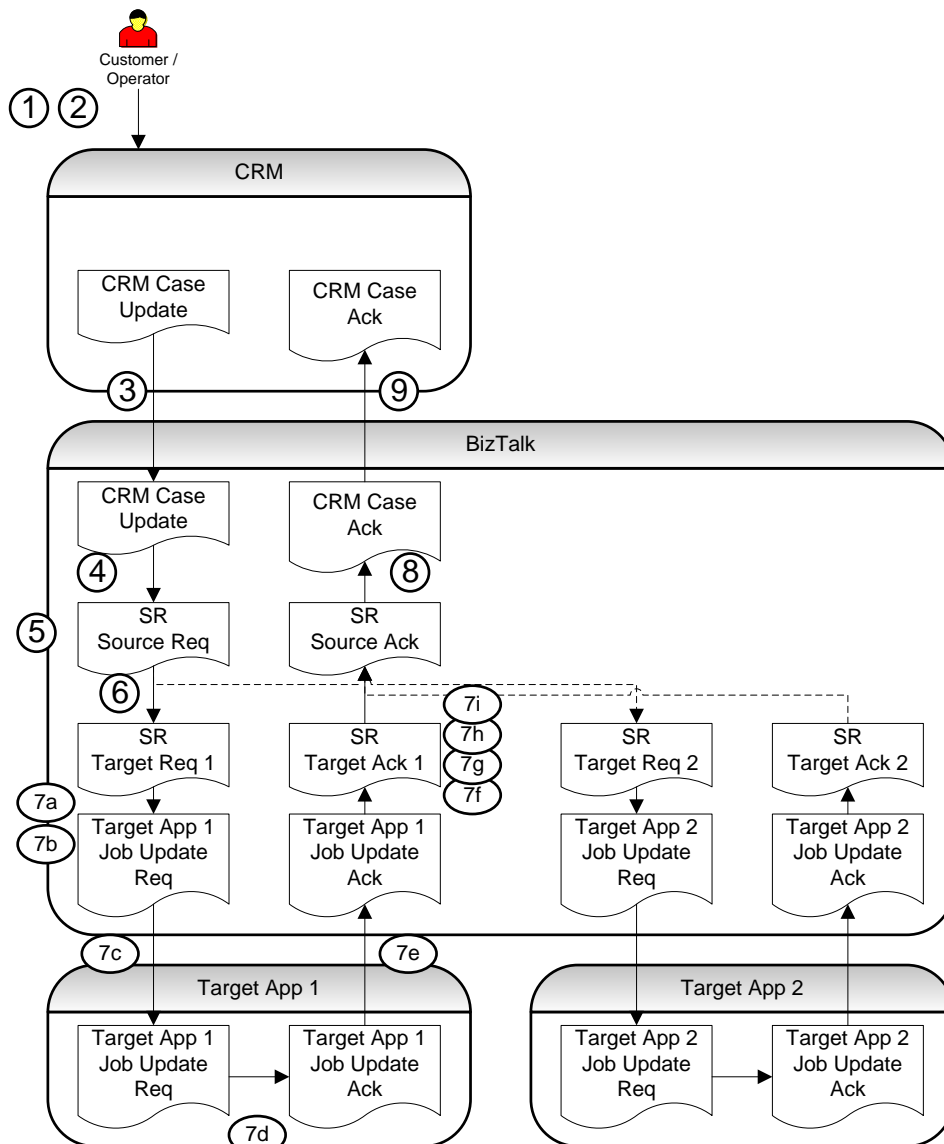
End

The use case ends when the updated details of the service request can be viewed in the appropriate Target Application Instance(s) and a success Acknowledgment has been returned to the CRM Application Instance.

Pre-requisites

- Customer has provided enough information to identify the Case in the CRM system.
- Any additional payments have been made and authorised (for CRM-based payment systems).

Basic Flow



1. The Customer contacts the council by email, phone, letter or fax, or through the self-service website.
4. An Operator (or the CRM interface, if through the self-service website) creates a Call in the CRM system which includes the contact details for the Customer, and updates the Case containing the details of the request.
5. The CRM API passes the Case details to BizTalk as a CRM Case Update Message, which creates a Service Request Message with a RequestType of "update" (see ref C for details of mandatory and optional fields).
6. As part of Service Request message creation, the system retrieves Common IDs that match CRM Local IDs for the Case, Customer, Location, Service and SubService.
7. The system validates the Service Request Message to ensure that all mandatory fields are present and all fields are of the correct data type.
8. BizTalk determines the list of Target Application Instances (other than the Source Application Instance) that have been configured to accept messages of this service type.
9. For each Target Application Instance in turn:
 - a. The Service Request Message is mapped to a **Job Update Message** for the Target Application Instance

- b. As part of message creation, the system retrieves **Local IDs** for **Case, Customer, Location, Service** and **SubService** for the **Target Application Instance**.
 - c. The Job Update Message is sent to the Target Application Instance API.
 - d. The Target Application Instance updates the Job.
 - e. The system receives a **Job Update Ack Message** from the **Target Application Instance** (see reference C for details of mandatory and optional fields). If the Target Application Instance cannot return a Target App Ack Message (because it is “fire and forget” eg email), steps 7f to 7i are skipped.
 - f. The Job Update Ack Message is mapped to a standard Ack Message.
 - g. The system validates the **Ack Message** to ensure that all mandatory fields are present and are of the correct data type.
 - h. The system takes the **Service Request Message** and the **Ack Message** and updates/creates Cross References according to **Business Rule 01**.
 - i. If the Source Application Instance is in the Target Application Instance’s Ack List, the AckID value is recorded.
10. The system maps the Ack Message to a CRM Ack Message, including the AckID and the JobID.
 11. The system sends the CRM Ack Message to the CRM Application Instance.

Alternate Flow AF1 - Service Request Message fails validation

1. If in BF step 5, the Service Request Message fails validation, the system records an error in the BizTalk application log.
2. The system constructs a CRM Ack Message, specifying the error in the ErrorDescription field.
3. The system returns the Ack Message to the CRM Application Instance.

Alternate Flow AF2 - Timeout error contacting Target Application Instance

1. If in BF step 7c a timeout error is received while trying to contact the Target Application Instance API, the system re-sends the message (in case it was a transient error). If the re-delivery is successful, flow resumes from step 7e.
2. If a timeout error occurs with the re-sent message, the system records an error the BizTalk server application event log and constructs a CRM Ack Message, specifying the error the ErrorDescription field and the Target Application Instance in the Error Source field
3. The system returns the Ack Message to the CRM Application Instance.

Note: The system does *not* attempt to deliver the message to any other Target Application Instances and does *not* attempt to “roll back” previous successful deliveries.

Alternate Flow AF3 - Target Application Instance returns error in Ack Message

1. If in step 7e the Target App Ack Message has a non-empty ErrorDescription, the system records an error in the BizTalk Server application event log.
2. The system constructs a CRM Ack Message, specifying the error the ErrorDescription field and the Target Application Instance in the ErrorSource field
3. The system returns the Ack Message to the CRM Application Instance.

Note: The system does *not* attempt to deliver the message to any other Target Application Instances and does *not* attempt to “roll back” previous successful deliveries.

Scenarios

- Update Abandoned Car request
- Update Vehicle Obstruction request
- Update Street Cleaning (Littering) request
- Update Fly Tipping request
- Update Dog Fouling request

- Update Bulky Waste request
- Update Missed Bin (Domestic) request
- Update Missed Bin (Commercial) request
- Update Stray Dogs request
- Update Dead Animal Removal request
- Update Recycling Bags request
- Update request – Service Request fails validation
- Update request – Target Application Instance unavailable
- Update request – Error from Target Application Instance

8.4.3 UC0103 Query Service Request

This use case covers an online query from CRM concerning the current status of a service. It assumes that case management is being done in the back-end system and not in the CRM system, i.e. the back end system is not “pushing” case updates to CRM.

Goal

Details of service request and job status are available in CRM.

Start

The use case starts when a “query service status” transaction is posted in CRM for a particular pre-existing service request by a Customer (via a self-service web site) or an Operator (via the CRM agent application).

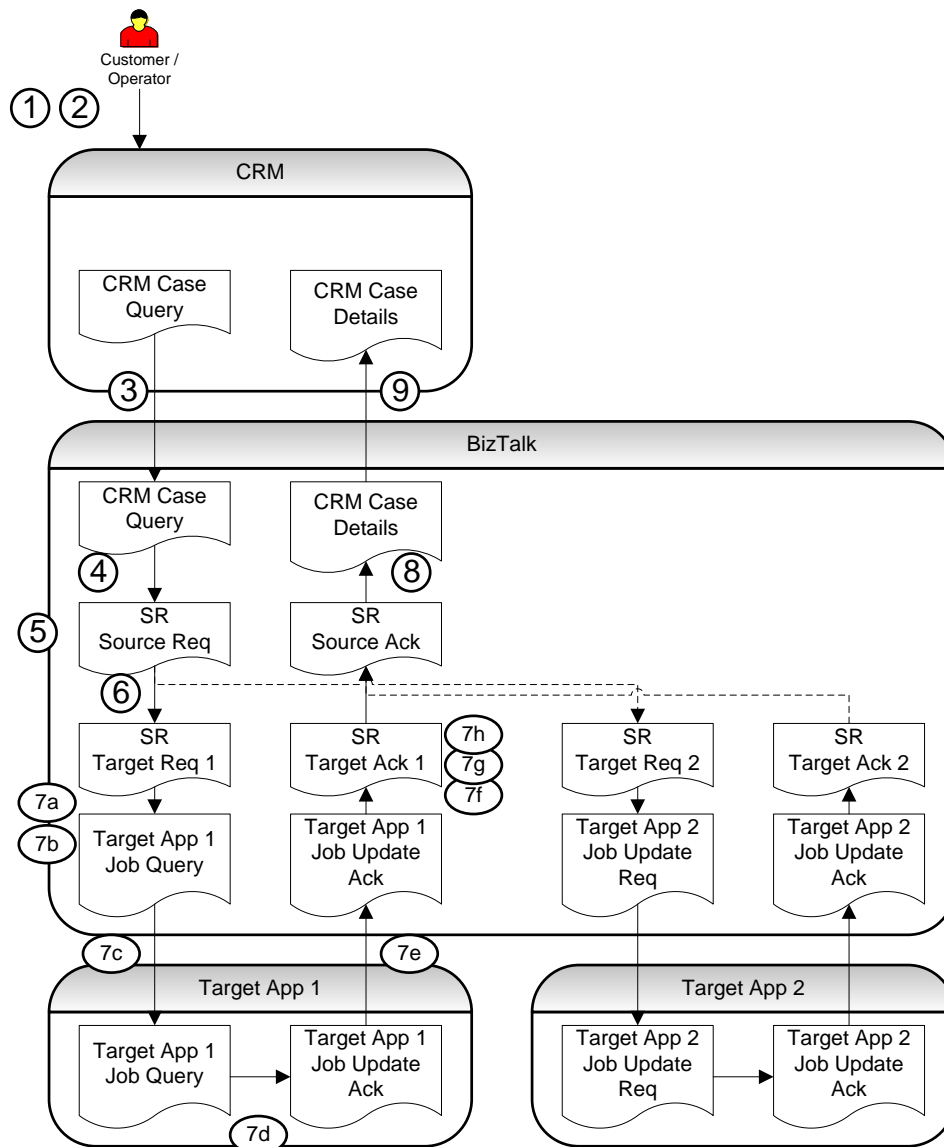
End

It ends when the current details of the service request from the back-end Application Instance are displayed in the CRM user interface, or are returned by CRM to the calling Application Instance.

Pre-requisites

- Customer has provided enough information to identify the Case uniquely
- Back-end Application Instance has a compatible query API.

Basic Flow



1. The Customer contacts the council by email, phone, letter or fax, or through the self-service website.
12. An **Operator** (or the CRM interface, if through the self-service website) creates a **Call** in the CRM system which includes the contact details for the Customer, and creates a **Query** that specifies the parameters needed by the **Target Application Instance** query API.
13. The CRM API passes the **Query** details to BizTalk, which creates a **Service Request Message** with a RequestType of "query" (see ref C for details of mandatory and optional fields).
14. As part of Service Request message creation, the system retrieves **Common IDs** that match CRM **Local IDs** for the **Case**, **Customer**, **Location**, **Service** and **SubService**.
15. The system validates the **Service Request Message** to ensure that all mandatory fields are present and all fields are of the correct data type.
16. BizTalk determines the **Target Application Instance(s)** that have been configured to accept messages of this service type and which have the **Source Application Instance** in their **Ack List**.
17. For each **Target Application Instance** in turn:
 - a. The Service Request Message is mapped to a **Query Message** for the Target Application Instance

- b. As part of message creation, the system retrieves **Local IDs** for **Case**, **Customer**, **Location**, **Service** and **SubService** for the **Target Application Instance**.
 - c. The Query Message is sent to the Target Application Instance API.
 - d. The **Target Application Instance** queries its data source to retrieve details of the **Case/Job**.
 - e. The system receives a **Job Details Message** from the **Target Application Instance** containing details of the **Case / Job**.
 - f. The **Job Details Message** is mapped to a standard **Ack Message** (see reference C for details of mandatory and optional fields).
 - g. The system validates the **Ack Message** to ensure that all mandatory fields are present and are of the correct data type.
 - h. If the **Source Application Instance** is in the **Target Application Instance's Ack List**, the JobID, Feedback Dialogue is recorded.
18. The system maps the **Target Application Instance Ack Message** to a **CRM Case Details Message**, including all details of the Case / Job.
 19. The system sends the **CRM Case Details Message** to the CRM Application Instance.

Alternate Flow AF1 - Target Application Instance cannot find Job/Case

1. If in step 7e the **Target Application Instance** returns an empty **Job Details Message**, the system constructs an **Ack Message** containing the details of the .
2. The system constructs a **CRM Case Details Message**, specifying the error in the ErrorDescription field.
3. The system returns the **Case Details Message** to CRM.

Alternate Flow AF2 - Service Request Message fails validation

1. If in BF step 5, the **Service Request Message** fails validation, the system records an error in the BizTalk application log.
2. The system constructs a **CRM Case Details Message**, specifying the error in the ErrorDescription field.
3. The system returns the **Case Details Message** to the CRM Application Instance.

Alternate Flow AF3 - Timeout error contacting Target Application Instance

1. If in BF step 7c a timeout error is received while trying to contact the Target Application Instance API, the system re-sends the message (in case it was a transient error). If the re-delivery is successful, flow resumes from step 7e.
2. If a timeout error occurs with the re-sent message, the system records an error the BizTalk server Application Instance event log and constructs a CRM **Ack Message**, specifying the error the ErrorDescription field and the Target Application Instance in the Error Source field
3. The system returns the **Ack Message** to the CRM Application Instance.

Note: The system does *not* attempt to deliver the message to any other Target Application Instances and does *not* attempt to “roll back” previous successful deliveries.

Scenarios

- Abandoned car query (“Why hasn’t the car I reported a week ago been removed?”)
- Street cleaning query (“There’s still broken glass on the pavement”)
- Bulky waste query (“It’s an hour past the appointment time and they still haven’t turned up”)
- Missed bin query (“My bins still haven’t been collected”)
- Pests query (“Could you remind me when my appointment was?”)

8.4.4 UC0104 “Push” Service Request Update

This use case covers the sequence of actions where a Department Employee updates a **Job/Case** in a back office Application Instance (the “Target Application Instance”) which then delivers it to CRM

(and possibly other target Application Instances). It only occurs if CRM is configured to accept “pushed” Case Updates.

The following updates are covered:

- Change in job/case status
- Change in job/case dialogue or hidden comments dialogue
- Assignment of officer
- **Note:** This use case does *not* cover creation of a new Case in CRM by creation of a job in the back office Application Instance (eg where a Customer has contacted the environmental services department directly rather than through the contact centre). The Case must already exist in CRM and must be cross-referenced.

Goal

The update to the Job / Case is visible in the target Application Instances. The Source Application Instance has received an Acknowledgment from CRM (optional).

Start

The use case starts when a back office Application Instance is updated. The backend Application Instance update needs to be transmitted to the case within the CRM system (assuming that CRM is handling case management).

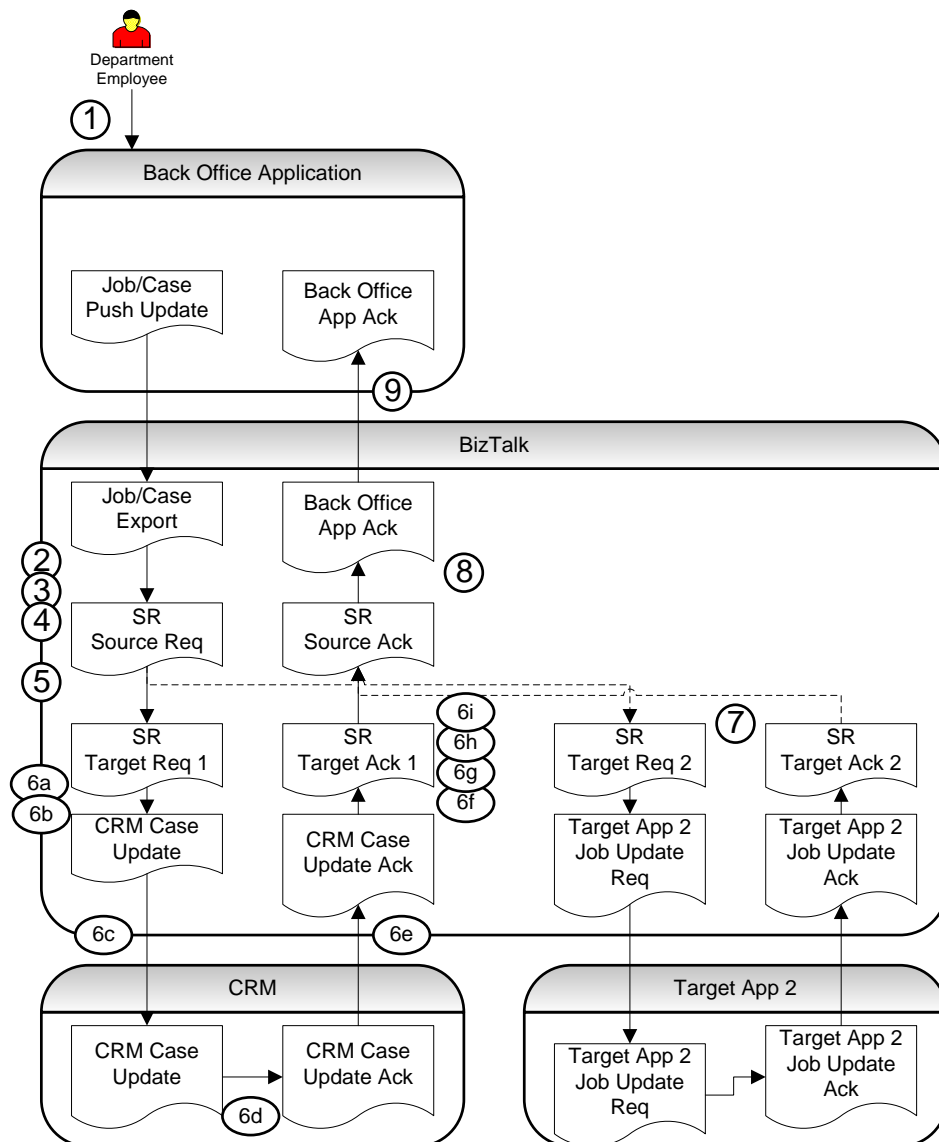
End

It ends when the update is visible in the Case notes in the CRM Application Instance and in other target Application Instances.

Pre-requisites

- Back-end Application Instance has a compatible “Export updates” API.
- Cross Reference exists allowing the CRM case to which the update refers to be identified, either as part of the exported message or from a queryable Cross Reference service.
- CRM can accept “pushed” Case updates.
- Case already exists in the CRM Application Instance.

Basic Flow



1. A Department Employee updates a job/case in the **Source Application Instance**, which submits a **Job Push Update Message** to BizTalk (NB it is assumed that the Source Application Instance can distinguish between job updates made through its UI and those made through its import API, and will only export the former).
1. BizTalk maps the Job Push Update Message to a Service Request Message with a RequestType of "update".
2. As part of **Service Request Message** creation, the system retrieves **Common IDs** that match Source Application Instance **Local IDs** for the **Case, Customer, Location, Service** and **SubService**.
3. The system validates the **Service Request Message** to ensure that all mandatory fields are present and all fields are of the correct data type.
4. BizTalk determines the list of **Target Application Instances** (other than the **Source Application Instance**) that have been configured to accept messages of this service type.
5. When CRM is the **Target Application Instance**:
 - a. The Service Request Message is mapped to a CRM Case Update Message.
 - b. As part of message creation, the system retrieves Local IDs for Case, Customer, Location, Service and SubService for CRM.
 - c. The CRM Case Update Message is sent to the CRM API.

- d. The CRM API updates the Case in CRM.
 - e. Optionally, The system receives a CRM Ack Message from CRM.
 - f. The CRM Ack Message (or the Service Request Message, if no Ack was returned) is mapped to a standard Ack Message.
 - g. The system validates the Ack Message to ensure that all mandatory fields are present and are of the correct data type.
 - h. The system takes the Service Request Message and the Ack Message and updates/creates Cross References according to Business Rule 01.
 - i. If the Source Application Instance is in CRM's Ack List, the AckID value is recorded.
6. The message is delivered to any other **Target Application Instances** as per UC0102 BF Step 7.
 7. The system maps the Service Request Message to a Source Application Instance Ack Message, including the AckID. If the **Target Application Instance** cannot return a **Target App Ack Message** (because it is "fire and forget" eg email), steps 7f to 7i are skipped.
 8. The system sends the Ack Message to the **Source Application Instance**.

Alternate Flow AF1 - Service Request Message fails validation

1. If in BF step 4, the **Service Request Message** fails validation, the system records an error in the BizTalk Application log.
2. The system constructs a Source Application Instance **Ack Message**, specifying the error in the ErrorDescription field.
3. The system returns the **Ack Message** to the Source Application Instance.

Alternate Flow AF2 - Timeout error contacting Target Application Instance (including CRM)

1. If in BF step 6c or 7 a timeout error is received while trying to contact the Target Application Instance API, the system re-sends the message (in case it was a transient error). If the re-delivery is successful, flow resumes from step 6e.
2. If a timeout error occurs with the re-sent message, the system records an error the BizTalk server Application Instance event log and constructs a **Source Application Instance Ack Message**, specifying the error the ErrorDescription field and the Target Application Instance in the Error Source field
3. The system returns the **Ack Message** to the **Source Application Instance**.

Note: The system does *not* attempt to deliver the message to any other Target Application Instances and does *not* attempt to "roll back" previous successful deliveries.

Alternate Flow AF3 - Target Application Instance returns error in Ack Message

1. If in BF step 6e or 7 the **Target App Ack Message** has a non-empty ErrorDescription, the system records an error in the BizTalk Server application event log.
2. The system constructs a **Source Application Instance Ack Message**, specifying the error the ErrorDescription field and the Target Application Instance in the ErrorSource field
3. The system returns the **Ack Message** to the **Source Application Instance**.

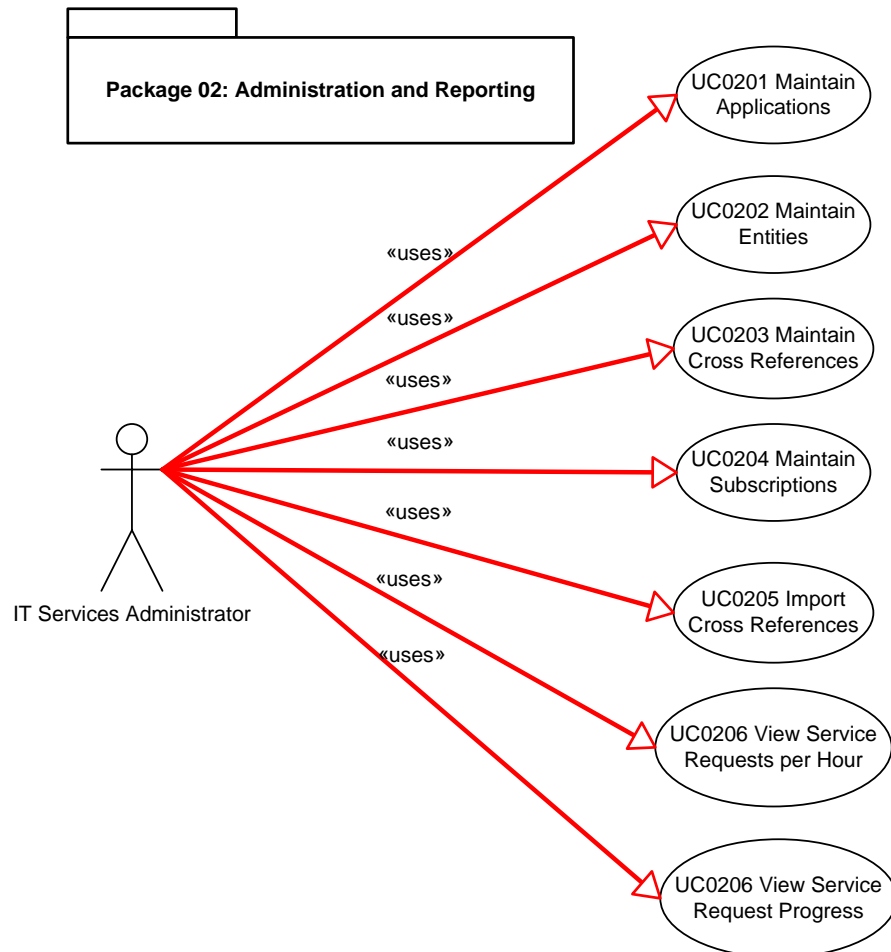
Note: The system does *not* attempt to deliver the message to any other Target Application Instances and does *not* attempt to "roll back" previous successful deliveries.

Scenarios

- Service status update ("Operator assigned", "Bins have been collected", "Street cleaned")
- Contractor reports problem ("Could not find the property", "attacked by dog when trying to access property")
- Environmental Services comment ("Customer has not paid their last invoice", "this is not a request that can be dealt with by us")

8.5 Package 02: Application Administration and Reporting

This package covers use cases relating to the administration of the solution.



8.5.1 UC0201 Maintain Application Instances

This use case covers the sequence of actions involved in maintaining the application instances supported by the integration hub.

Note: This use case is based on the capabilities of the built-in BizTalk cross-referencing system, which has the concepts of Applications and Application Instances. An “Application” in the Toolkit Domain Model corresponds to a BizTalk “Application Instance”.

The following updates are covered:

- Add Application
- Remove Application

Goal

Application updated in the BizTalk management database and can be referenced by the cross referencing and subscriptions systems.

Start

The use case starts when an IT Services Administrator needs to update the list of applications supported by the integration hub.

End

The use case ends when an application instance has been added to or removed from the integration hub list.

Pre-requisites

- User is logged in with a Windows account
- User has permissions to access the administration UI.

Basic Flow - Add Application Instance

1. The user starts the administration UI.
2. The system displays the list of **Applications** that are currently supported as a pageable list in alphabetical order, with a text box for entry of new Applications. The following details are shown for existing applications: Application ID, Application Name
3. The user enters the **standard name** of the **Application** and clicks the Save button.
4. The system checks that the standard name does not already exist and creates a new database record, assigning an ID. It reshows the Applications list with the new name.
5. If the standard name already exists, the system shows a “Duplicate name” error to the user. *Use case ends.*
6. The user selects the new **Application** by clicking its “Select” button.
7. The system shows an empty box for the name of the **Application Instance**.
8. The user enters the **standard name** of the **Application Instance** (the same as the **Application** name if there is only one instance in use) and clicks the Save button.
9. The system creates a new database record for the **Application Instance** and assigns an ID. It reshows the list of Application Instances including the new Application.
10. The user repeats steps 5 to 8 to create any other instances of the application.

Alternate Flow AF1 - Delete Application

Pre-requisite: All Cross References and subscriptions relating to the Application Instances to be deleted have been removed (see Use Cases below).

1. In BF step 3, the user selects the Application to be deleted by clicking on its Select button.
2. The system shows the IDs and names of the Application Instances associated with the Application as a pageable list.
3. For each Instance of the Application to be deleted:

- a. The user clicks on the Application Instance Delete button
 - b. The system shows an “Delete this record?” message with Yes and No buttons.
 - c. The user clicks the Yes button to confirm the deletion. If the user clicks No, use case ends.
 - d. The system checks that all dependent records for the Application Instance have been deleted, deletes the Application Instance record and reshows the Application Instances list.
9. The user clicks the Delete button for the Application to be deleted.
 10. The system shows a “Delete this record?” message
 11. The user clicks the Yes button to confirm the deletion. If the user clicks No, use case ends.
 12. The system checks that no dependent records exist, deletes the Application record and reshows the Applications list.

Alternate Flow AF2 - Dependent records exist for a record to be deleted

1. If in AF1 step 3d or AF1 step 7 the system determines that a dependent record exists, it displays an error to the user informing them that a dependent record exists and advising them to check the cross-references and subscriptions (for an Application Instance), or the Application Instances (for an Application). Use Case ends.

Scenarios

- Add Application
- Add Application – duplicate standard name for Application
- Add Application - duplicate standard name of Application Instance
- Delete Application
- Delete Application - dependent records exist – Application Instance.
- Delete Application – dependent records exist - Application

8.5.2 UC0202 Maintain Entities

This use case covers the sequence of actions involved in updating the list of cross-referenceable Entities (Cases, Customers, Locations etc) in the system

Goal

A cross-referenceable Entity is added to or removed from the system.

Start

The use case starts when an IT Services Administrator needs to add a new Entity type to or remove an Entity type from list of the cross-referenceable Entities supported by the system.

End

The use case ends when the Entity has been added or removed.

Pre-requisites

- User is logged in with a Windows account
13. User has permissions to access the administration UI.

Basic Flow - Add Entity

1. The user starts the administration UI.
2. The system displays the IDs and **standard names** of the cross-referenceable **Entities** as a pageable list, in alphabetical order, with a select and delete buttons for each row and a text box for new entities.
3. The user enters the **standard name** of the new Entity and clicks the Save button.
4. The system checks that the standard name is unique, creates the record assigning a unique ID and re-shows the list of Entities.

5. If the standard name is not unique, the system displays a “duplicate name” error to the user.

Alternate Flow AF1 - Delete Entity

1. In BF step 3, the user clicks the Delete button for the **Entity** to be deleted.
2. The system displays a “Delete this record?” message.
3. The user clicks “Yes” to confirm deletion of the Entity. If the user clicks “No”, *use case ends*.
4. The system checks that no dependent record exists, deletes the Entity record and re-shows the Entity list.
5. If a dependent record exists, the system displays a “Dependent record exists” error to the user and advises them to check that all cross-references and subscriptions for the entity have been deleted.

Scenarios

- Add Entity
- Add Entity – duplicate standard name
- Delete Entity
- Delete Entity – dependent records exist.

8.5.3 UC0203 Maintain Cross References

This use case covers the sequence of actions involved in searching for, adding or removing a single Cross Reference. Note: This use case does *not* cover the batch addition or removal of cross-references.

Goal

Cross Reference added to or removed from the system.

Start

The use case starts when an IT Services Manager determines that a manual change has to be made to a Cross Reference, either because it has been wrongly assigned due to a system error or because it was never created when it should have been.

End

The use case ends when the Cross Reference change has been made and saved by the system.

Pre-requisites

- User is logged in with a Windows account
- User has permissions to access the administration UI.
- **Application Instance** and **Entity** exist for the Cross Reference to be added.

Basic Flow BF - Search by Local Identifier

This search is used to identify the **Common Identifier** associated with an **Application Instance’s Local Identifier**.

1. The user starts the Administration UI.
2. The system displays IDs and **standard names** of the **Applications** and the cross-referenceable **Entities** available in the system as pageable lists, with Select buttons.
3. The user selects the Application to Search by clicking its Select button.
4. The system displays the IDs and Names of **Application Instances** associated with the selected Application as a pageable list, with Select buttons.
5. The user selects the **Application Instance** to search by clicking its Select button.
6. The user selects the **Entity** to search by clicking its Select button.

7. When both an **Application Instance** and an **Entity** have been selected by the user, the system displays a search text box for the **Local Identifier** for the Cross Reference.
8. The user enters the first few letters of the **Local Identifier** (up to 10?).
9. The system displays the first 50 **Local Identifiers** matching the entered characters.
10. The user selects the **Local Identifier** to be searched on. If the Local Identifier is not in the list, the user changes their search term or cancels (*use case ends*).
11. The system displays the **Common Identifier** matching the **Local Identifier**.

Alternate Flow AF1 - Search by Common Identifier

This search is used to identify all the **Local Identifiers** associated with a particular **Common Identifier**.

1. In BF step 3, the user clicks the “Search by Common Identifier” menu option.
2. The system displays a text boxes for the **Entity** and **Common Identifier**.
3. The user enters the first few letters of the **Entity**.
4. The system displays the first 50 **Entity Standard Names** matching the entered characters
5. The user selects the Entity to be searched on.
6. The user enters the first few letters of the **Common Identifier** to be searched on (up to 10?).
7. The systems displays the first 50 **Common Identifiers** matching the entered characters.
8. The user selects the **Common Identifier** to be searched on. If the desired Common Identifier is not the list, the user changes their search term or cancels (*use case ends*).
9. The system displays the **Application Instance** and value for all **Local Identifiers** associated with the Common Identifier as a pageable list.

Alternate Flow AF2 - Add Cross Reference

1. In BF step 6, the user clicks on the “Add New Cross Reference” button.
2. The system displays text boxes for both **Local Identifier** and **Common Identifier**.
3. The user enters the **Local Identifier** for the selected **Application Instance** and **Entity**.
4. The user enters the first few letters of the associated **Common Identifier** (if it already exists) *or* the entire **Common Identifier** (if it does not), and clicks “Add”.
5. The system checks to see if there are any **Common Identifiers** that start with the text that the user entered. If there is not, the system creates the Cross Reference for the specified **Application Instance, Entity, Local Identifier** and **Common Identifier** and displays a “Cross Reference created” message to the user. *Use Case ends*.
6. If one or more **Common Identifiers** match the text that the user entered, the system displays the first 50 matches.
7. The user selects the **Common Identifier** to use. If the required **Common Identifier** does not appear, the user changes their search term (return to step 5) or cancels (*use case ends*).
8. The system creates the Cross Reference for the specified **Application Instance, Entity, Local Identifier** and **Common Identifier** and displays a “Cross Reference created” message to the user.

Alternate Flow AF3 - Delete Cross Reference

1. In BF step 11 *or* AF1 step 9, the system displays a Delete button next to each displayed Cross Reference.
2. The user clicks the Delete button next to the Cross Reference to be deleted.
3. The system displays a “Delete this record?” dialogue.
4. The user clicks Yes to confirm the deletion, or No to cancel (*use case ends*).
5. The system deletes the Cross Reference, displays a “Cross Reference deleted” message to the user and re-shows the list of Cross References (if appropriate).

Scenarios

- Search for Cross Reference for Local Identifier
- Search for all Cross References for Common Identifier
- Add Cross Reference
- Delete Cross Reference – Local Identifier search screen
- Delete Cross Reference – Common Identifier search screen

8.5.4 UC0204 Maintain Subscriptions

This use case covers the sequence of actions involved in adding, editing or deleting a **Subscription** by an **Application Instance** to a **Service**.

Goal

Application Instance added to or removed from the list of subscribing Application Instances for a Service/SubService.

Start

The use case starts when an IT Services Administrator wishes to change the list of Application Instances that can publish or subscribe to service requests for a particular Service/SubService.

End

The use case ends when the system confirms that the subscription has been added, edited or deleted.

Pre-requisites

- User is logged in with a Windows account
 - User has permissions to access the administration UI.
14. Application Instance exists (see use case above)

Basic Flow BF - View Subscriptions

1. The user starts the administration UI and selects the “SLG Subscriptions” link.
2. The system displays text boxes for **Application Instance** and **Service** search controls and the first 50 Subscriptions as a pageable list (max 50 records per page) with the following fields: **Application Instance**, **Service**, **SubService**, **Ack List**, ApplInstance Service Code, ApplInstance SubService Code. The system also displays Edit and Delete buttons for each record and controls for entering fields for a new entry.
3. The user enters the first few characters of the **Application Instance** or **Service**.
4. The system displays the first 50 standard names matching the Application Instance or Service search term.
5. The user selects the **Application Instance** or **Service** to search on.
6. The system redisplay the pageable list, filtering it by **Application Instance** or **Service**.

Alternate Flow AF1 - Add New Subscription

1. In BF step 2, the user selects the **Application Instance** in the “New Application Instance” control.
15. The user enters the first few letters for the **Service** in the “New entry” Service text box.
16. The system displays the first 50 **Service standard names** that match the entered characters.
17. The user selects the **Service** to be used, or types in a new **standard name** for the **Service**.
18. The system populates the **SubService** control with **SubService** values for the service (if any), setting the control default value to “All SubServices”.
19. If the **Application Instance** should subscribe only to a particular **SubService**, the user selects the **SubService** to be subscribed to.

20. If the subscribing **Application Instance** has an **Ack List**, the user enters the **standard name(s)** of the **Application Instance(s)** that the subscribing **Application Instance** should return an **Ack** to.
21. The user enters values for the AppInstance Service Code and, if appropriate, the App Instance SubService code, and clicks the Save control.
22. The system checks that a **Subscription** for the **Service, SubService** and **Application Instance** does not already exist and creates the **Subscription**. If the Subscription already exists, the system displays a “Subscription exists” error message - *use case ends*.
23. The system creates or updates **Cross Reference** records for the **Service (Common Identifier)** and App Instance Service Code (**Local Identifier**), and, if appropriate, for the **SubService (Common Identifier)** and App Instance SubService Code (**Local Identifier**). The system displays a message that the **Subscription** was successfully created and updates the pageable list.

Alternate Flow AF2 - Delete Subscription

1. In BF step 6, the user selects the record to delete by clicking its Delete button.
24. The system displays a “Delete this record?” dialog.
25. The user clicks Yes to confirm the deletion, or No to cancel (*use case ends*).
26. The system deletes the Subscription and the associated **Cross References** for the **Service** and optionally **SubService** for the **Application Instance**.

Scenarios

- Search for Subscriptions for Application
 - Search for Subscriptions for Service
 - Add Subscription
 - Add Subscription – Subscription exists
27. Delete Subscription

8.5.5 UC0205 Import Cross References

This use case covers the sequence of actions involved in doing a batch import of cross references from a flat file.

NOTE: Requirement not agreed. Do not implement yet.

Goal

New and updated Cross References imported into the Cross Reference system.

Start

The use case starts when an IT Services Administrator wishes to import a flat file of cross references.

End

The use case ends when the system reports that the import process is complete.

Pre-requisites

- User is logged in with a Windows account
- User has permissions to access the administration UI.
- Application Instance exists

Basic Flow

1. The user starts the Administration UI and selects the “Import Cross References” option.
2. The system displays a list of available **Application Instances** and **Entities**.
3. The user selects the **Application Instance** and **Entity** that the import is for.
4. The system displays a browse dialog to allow the user to select the import file.
5. The user selects the file to import.

6. The system checks that the file format is valid.
7. For each row in the file:
 - a. The system checks that the **Local Identifier** is unique for the **Application Instance** and **Entity**. If it is not, it appends the row in error and the error message to an error log file.
 - b. The system checks that the **Common Identifier** does not already appear in a **Cross Reference** for the **Application Instance** and **Entity**. If it does, it appends the row in error and the error message to an error log file.
 - c. The system creates the Cross Reference for the specified Application Instance, **Local Identifier** and **Common Identifier**.
8. The system displays the number of rows that were successfully imported and the number of errors.

Scenarios

28. Import cross reference file – no errors
29. Import cross reference file – errors.

Outstanding issues

30. In step 7b, should a pre-existing cross-reference for a Common Identifier simply be deleted, ie allow update of cross references? Or should it be configurable?

8.5.6 UC0206 View Service Requests Per Hour

This use case covers the sequence of actions involved in viewing a report of the number of service requests received per hour.

NOTE: Requirement not agreed. Do not implement yet.

Goal

Determine the number of service requests per hour for a particular Service and a total number of service requests per hour.

Start

The use case starts when the user wants to view a report of the number of service requests per hour for a particular time period.

End

The use case ends when a report showing the appropriate data is available to the user on the screen.

Pre-requisites

- User is logged in with a Windows account
- User has permissions to access the administration UI.

Basic Flow

1. The user starts the administration UI and selects the “Service Requests by Hour” link
2. The system displays a pivot table and chart.
3. The user selects the year(s), month(s), day(s), hour(s) and minute(s) for which they want to see requests and optionally the service type(s) to use.
4. The system displays the number of requests received for each service type for the specified time periods as a table and as a chart.

Scenarios

- All service requests in a 24 hour period

- Number of Missed Bin requests in a 24 hour period

8.5.7 UC0207 View Service Request Progress

This use case covers the sequence of actions involved in tracking the progress of a service request through the system.

NOTE: Requirement not agreed. Do not implement yet.

Goal

A report listing the messages that have been received for a particular Case and their associated Case status (open, closed, cancelled).

Start

The use case starts when a user wishes to determine the progress of a particular case.

End

The use case ends when the system displays a report showing the following information for all messages relating to the Case: Date/time received, message type, Case status, Error description (if any).

Pre-requisites

- User is logged in with a Windows account
- User has permissions to access the administration UI.
- User knows the Local ID in the CRM system for the Case to be tracked.

Basic Flow

5. The user starts the administration UI and selects the “Find Service Request” link
31. The system displays a text box for the Case Local ID.
32. The user enters the Case Local ID.
33. The system displays a table of messages with Case Common IDs that match the specified Case Local ID, showing Date/time received, message type, Case status and Error description (if any).

Scenarios

-

9 Business Rules

9.1 Business Rule 01: Cross Reference Creation

This rule describes the logic for creating or updating cross references once a message has been processed by the Target Application Instance.

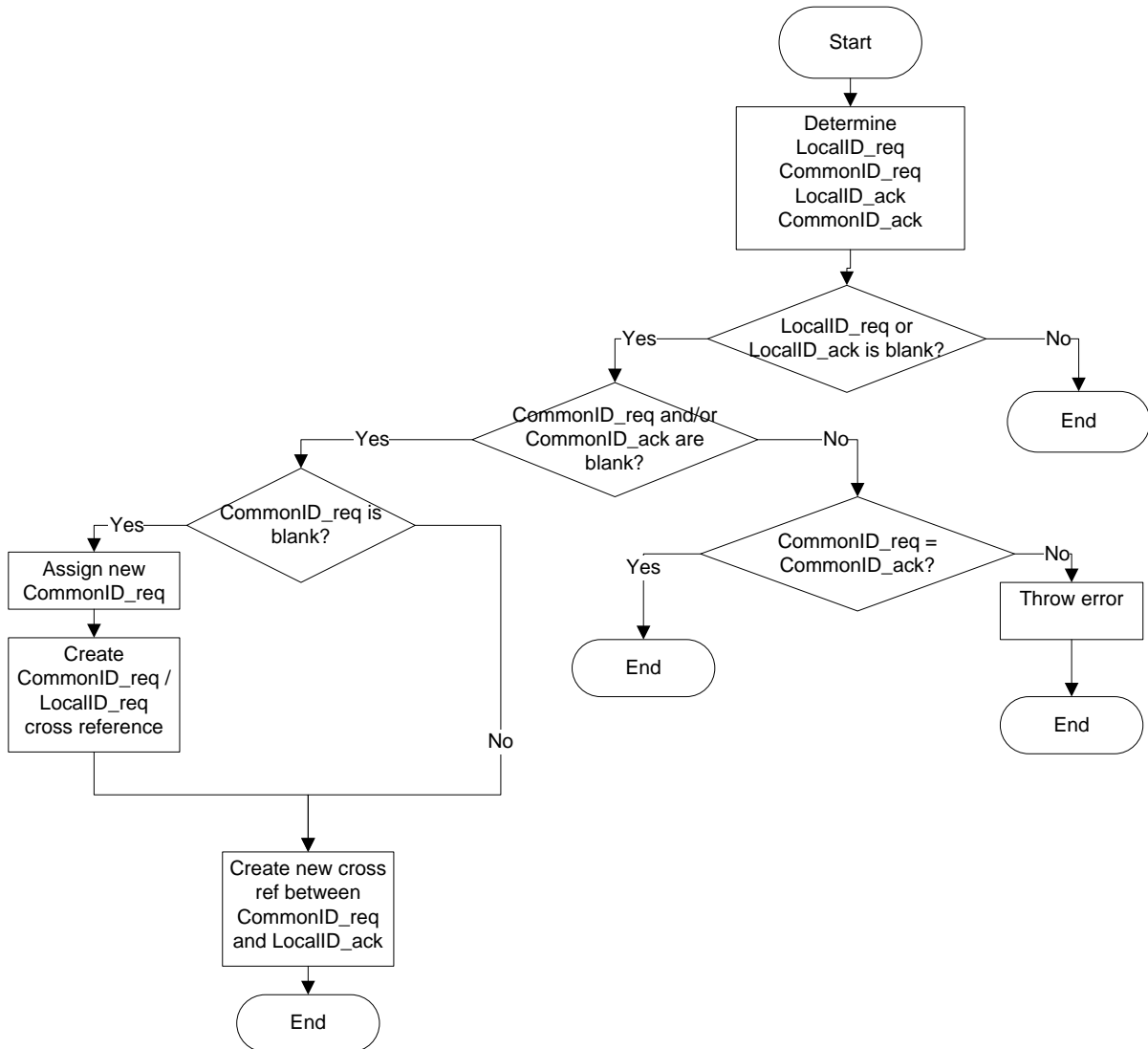
Inputs:

- Target Application Request message (req)
- Target Application Ack message (ack)
- List of Identifiers to be mapped

For each Identifier to be mapped:

- a. Determine Entity type
- b. Determine req CommonID value (CommonID_req)
- c. Determine req LocalID value (LocalID_req)
- d. Determine ack CommonID value (CommonID_ack)
- e. Determine ack LocalID value (LocalID_ack)

f. Create cross references according the flow chart below:



10 Roles and Security

This section describes the people, their roles and levels of security afforded each in the new application.

10.1 Roles

The following roles have been defined:

- System Administrator
- IT Services Administrator
- Application Users (CRM and back office)

10.2 Security

The following table summarises the access permissions required:

Function	System Admin	IT Services Administrator	Application Users
Administration UI (including reports)	Y	Y	N
Application interfaces	Y	Y	Y

11 Data Requirements

This section covers how data will be represented.

11.1 Existing Data Definitions

To be covered by application interface specifications.

11.2 Data Conversion

To be covered by application interface specifications.

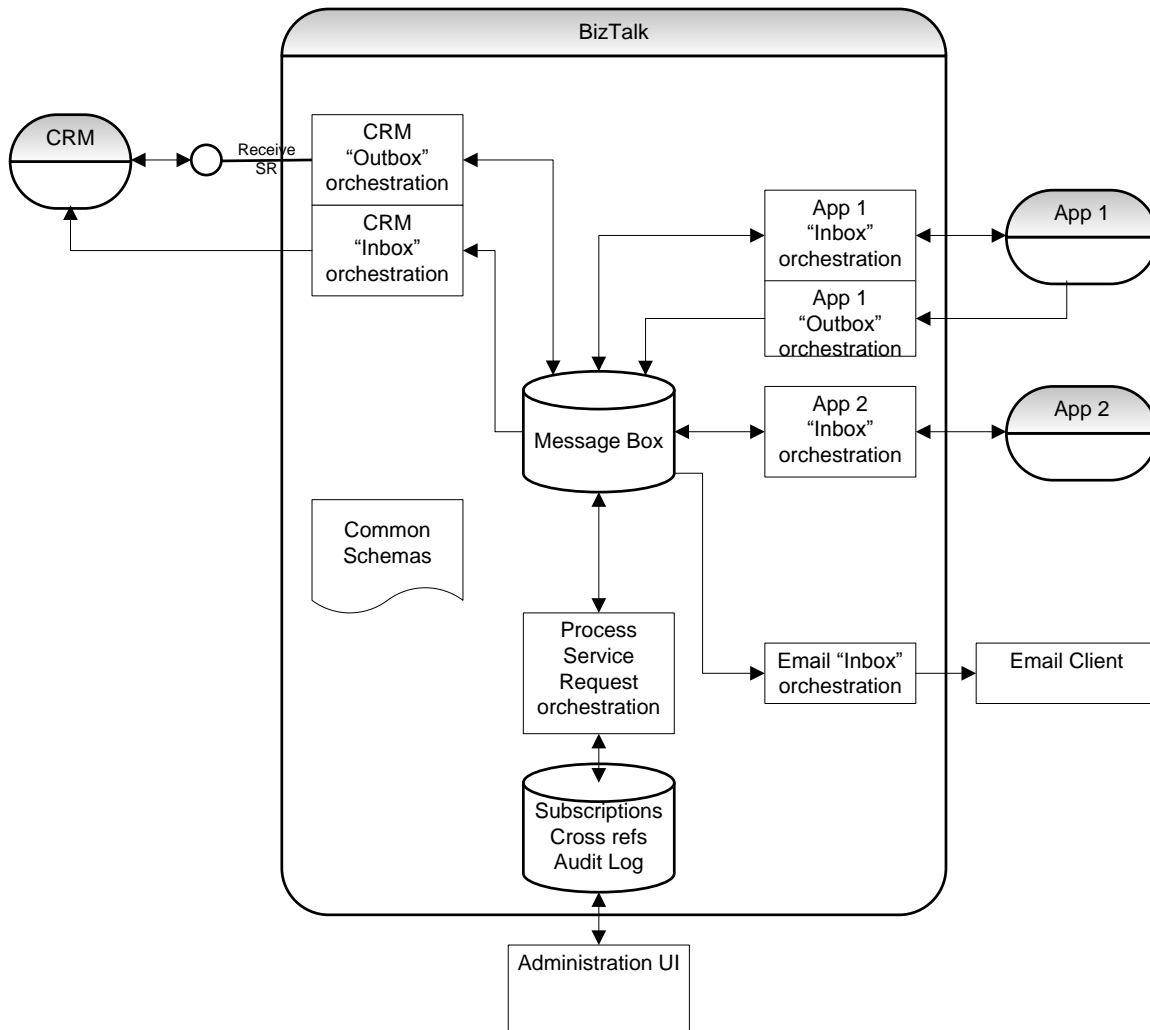
12 Software Architecture Overview

This section provides a brief architectural overview of the system, using different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system during the discover phase and will be detailed further during the design phase.

12.1 Logical View

This subsection describes the significant design packages and subsystems at this stage. During the design phase, the software architecture will be developed further to provide the 'blueprints' for the develop phase.

The following diagram summarises the current thinking about the architecture of the orchestration:



The chief features are:

1. Loose coupling of applications using the BizTalk message box as a central message drop
2. A set of common schemas to and from which all application-specific messages are mapped.
3. A core "Process Service Request" orchestration that handles the following:
 - Service request tracking
 - Determination of target applications for specific service types
 - Creation of cross references
 - Creation and delivery of source app acknowledgements
4. "Inbox" orchestration for each application (including CRM) that handles the following:
 - Mapping of common schemas to application-specific messages

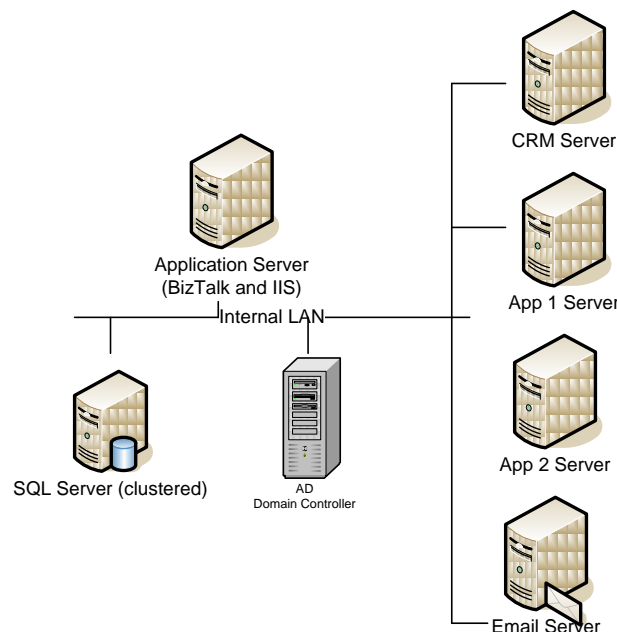
- Delivery of messages from the message box to application APIs (including batching/debatching and call sequences, if necessary)
 - Delivery of acknowledgments from application APIs to the message box
 - Error handling and retries
5. “Outbox” orchestration for each application (including CRM) that handles the following:
- Acceptance of messages from application APIs (including batching/debatching and call sequences, if necessary) for transactions which have started in the application
 - Mapping of application-specific messages to common schemas
 - Delivery of acknowledgments from the message box to source application APIs
 - Error handling and retries
6. Administration UI covering:
- Set up of Applications
 - Set up of Cross-referenceable Entities
 - Manual maintenance of cross references
 - Set up of Application subscriptions to Services
 - Viewing of reports on service requests per hour and status of individual Cases.

Cross reference mapping will use a “CommonID / LocalID” model where LocalIDs for pre-existing entities are mapped to CommonIDs as messages come into the system and CommonIDs are mapped to application LocalIDs as messages exit the system. Creation of new cross references needs to be centrally handled so that responses from individual applications can be mapped to the source service request that generated them. The alternative approach would be to handle all cross referencing in the central service request processing orchestration, but this would be an inflexible solution.

12.2 Deployment View

The following diagram shows a possible deployment configuration for the solution. It is based on the following assumptions:

- Minimal number of physical boxes
- Disaster recovery period of up to 24 hours is acceptable
- None of the back-office applications is on a remote network



The Application Server hosts all the BizTalk and web service components of the solution as well as the administration UI. The CRM and servers for the back office applications would host the APIs for those applications.

To increase resilience, a second application server could be introduced running a second instance of the BizTalk host. This would avoid the application server being a single point of failure.

If necessary, existing BizTalk and SQL Server instances could be used, provided that performance bottlenecks are not likely to occur.

13 Risks & Constraints

This section covers project risks and constraints

13.1 Technical Risks

Risk	Impact	Mitigation
Supplier APIs could severely limit integration	High	Obtain as much information as possible on supplier APIs. Develop relationships with suppliers to develop adequate APIs. If necessary, use direct integration to application datastores or submit updates as emails for manual entry.
No standard entity definitions for services, service requests, customers and locations. Mapping to the equivalent entities in back-end applications could be complicated.	Medium	Ensure that plenty of time is allowed for development interfaces. Make clear to suppliers what our understanding of these entities is with good documentation.
All components of the solution depend on the common schemas. Changing them to respond to particular integration requirements would force a recompile and reinstall of the entire solution.	Medium	Allow for multiple concurrent versions of schemas and core components.
The definition of the Case entity is ambiguous. Depending on where Case de-duplication occurs, it could either correspond to a contact from a particular customer, or a job implemented to satisfy multiple customer requests, or something in between.	Medium	Make sure that what a Case represents is agreed up front for each implementation.
Reporting requirements are not properly defined.	Low	Agree requirements up front.

13.2 Schedule Risks

None identified so far.

13.3 Constraints

Constraint	Consideration
Microsoft technology should be used.	-

14 Training Needs

This section describes the training requirements for the solution

None identified so far.

15 Project Approach

15.1 Project Management

Project management will be shared between CIBER and SLG. Responsibilities will be as follows:

Task	Responsibility
Approval and sign off of project requirements and design	SLG
Development of project plan	CIBER
Resourcing of developers	CIBER
Cost tracking and budgetary approval	SLG
Code development monitoring	CIBER
Change management	CIBER
Bug tracking	CIBER
Provision of system test environment	SLG
Deployment to system test environment	CIBER
User Acceptance Testing	SLG
Training and handover	CIBER

15.2 Time Boxing

The project will adopt a time boxed approach to enable the project team to deliver sufficient functionality to enable the web site to be deployed within the chosen timeframe. Under time boxing, the project team undertake to deliver an operational system by the target date in return for a complementary undertaking by the client that the result will be deemed satisfactory if a minimum usable subset of functionality is produced. Although plans will be drafted with the intention of producing more than this minimum subset, functional scope can be reduced by the customer and the project team in order to assure delivery in spite of negative incidents that impact the project. The minimum usable subset has been defined by means of the Requirements Catalogue and is represented by "Must" priority requirements. The following prioritisation scheme will be applied to the requirements

1	Must	Represents very important functionality that is required for launch.
2	Should	Represents important additional functionality but there are manual work-arounds or this functionality can be added later
3	Could	Represents those requirements that are useful and provide some degree of benefit.

16 Estimates

16.1 Assumptions

In order to provide the following estimates it has been necessary to make the following assumptions:

- Core components and test harnesses only – excludes development of application-specific components
- Queries will be handled in the same way as service requests, ie separate orchestrations will not be needed.
- No significant changes to functionality will result from the prototype review.
- Deployment to pilot environment is not included.

It is expected that during Design these assumptions will be removed from the project and replaced with fact.

16.2 Estimation

Note: The following estimates are provisional only and should not be used for detailed budgetary planning.

Task	Estimate (d)
Detailed Component Specification: Core Schemas	1
Detailed Component Specification: Core orchestrations and cross-reference / itinerary determination helper classes	1
Detailed Component Specification: Administration UI	1
Code and unit test core schemas	2
Code and unit test core orchestrations and helper classes	3
Code test harnesses	1
Code and unit test the Administration UI	5
Configure and test BAM components	1
Develop deployment packages	1
Deploy to system test environment	1
Write test cases	5
System test and fix	3
Documentation: Administration User Interface	1
Documentation: "How tos" and common errors document	1
Documentation: Solution installation guide	1
Documentation: Updates to design document and component specifications	2
Documentation: Application interface developer's guide	1
Documentation: Application interface specification template	1

Review and package toolkit	3
Project Management	5
TOTAL	40

17 Project Schedule

17.1 Milestones

The following milestones are suggested for development of core code component

Phase	Milestone
Design	Review prototype
	Approve and sign off Discover and Design documents
Develop	Core component code complete
	System testing complete
Deploy	Documentation pack complete
	Final sign-off of toolkit package

17.2 Project Plan

To follow.

18 Glossary

This section defines the acronyms and specialised technical and business terms used in this document. See also the domain model definitions above (Section 3).

API

Application Programming Interface.

BS7666

UK national standard for addresses.

BS8766

UK national standard for personal data such as names, telephone numbers, emails etc.

CRM

Customer Relationship Management. An application which tracks and organises information about customers and their contacts with an organisation.

KPI

Key Performance Indicator. A standard metric of the performance of a Local Authority in a particular area.

SLG

Shared Learning Group. A forum through which a number of local authorities are able to share ideas and best practice solutions.

Street Scene

Council services relating to street cleanliness and order, such as removal of abandoned cars, dead animals and litter, rounding up of stray dogs, collection of domestic and commercial waste etc.

UI

User interface.