# Building a System

## Business Definition

### Objectives

Definition of the objective of the project. Very high level defining the business impact and expected benefit. It may include overall expected timescales, costs, reliability, performance.

### Overview

Gives a description of what is available as inputs to the system and how they might be utilised.
It also defines the scope of the system including where the data originates, the items to be included and the required user interface.

### Constraints

This defines any external constraints on the system.
This can be items like technology to be used or expertise that is to be available.
Any issues about security of the data and system

### Initial plan

Make an attempt at a high level plan giving timescales and deliverables
This will mainly include gathering information about datasources and how they are to be used.

## Data Analysis

Gain access to the source systems and identify knowledge experts
Define what is required from each source system giving minimum and maximum expectations.
View data and record representations.
Note: the technology used here is unimportant as it may be changed. This may be a compromise between what is available, easiest and what is expected in the final product.
Any information gained should be recorded in the technical documents for future expansion

Prototype phase I

- Extract providing at least the minimum requirements for a restricted simple subset.
- Include only datasources that can be access easily.
- Transform data and merge datasources
- Check the result against the source systems or any independent representation
- Review definitions and possibly amend prototype

Prototype phase II

- Enumerate the known dataset subsets and how they can be obtained
- Extract samples for each subset and record issues
- Transform and merge data. Record and exclude depending on issues found
- Create a document of issues that need to resolved

Review phase

- Discuss issues found with the knowledge experts and record any potential solutions
- Discuss the result with the business with a view to identifying critical issues and maybe define a phased approach to the implementation, being careful to isolate functionality that is excluded in earlier phases.
- Define how issues will be resolved, timescales and resources needed.

Prototype phase III

- Provide a first cut of the complete system with a restricted set of data.
- Check product against other systems and the business users

## Technical Implementation

### Detailed plan

This will include all the sources and destinations, documentation, infrastructure, dependancies and development.
It should include a testing and implementation phase.
Note, this will be a live document and should be expected to change as the system progresses.

### Source Overview

Definition of each of the sources including
Stakeholders
Description of the type of data available including granularity
Data volumes
Data issues
    Any expected issues with the data e.g. data quality, reference data mismatches with other systems
Access - business
    When the source can be accessed and any issues with impact on the source
    Whether data can be pulled or needs to be delivered
    When data will be available e.g. due to updates
    Any times when the data cannot be accessed due to impact on other systems
Access - Technology
    Methods available/required to access data
    Security issues

### Destination Data Store Overview

The delivery point for the data.
This may be in several stages as the project progresses and will often include a staging area, data warehouse,  datamarts, cubes, reports.
It should give an idea of the platform and how the data will be held e.g. a data warehouse, relational, source copy.

### Destination Reporting Systems

User access to the system.
In the early stages this will probably be queries or simple reports to verify the data but as it grows will probably include descriptions of reports.

## Detailed definitions

A good start for the documentation is the Kimball definition spreadsheet. The data from which can be loaded into a metadata table and used to generate code.

### Source

This gives a definition of each datasource including the columns, datatypes and descriptions and possibly the cardinality. This may be the same as the staging area if the data is transformed and delivered externally, if so that transformation should be described.

### Staging

This gives a description of the staging area which may well be the same as the source data

### Data store definition

This describes each table and column in the destination and the source of that data. It should be based on the staging data and define the mapping between the staging and data store.

It should include a description of the datastore – relational, dimensional…
This description should be detailed enough that the data store and transformation processes can be generated from the definition.

### ETL objects

How data is transferred from the source to the destination.
Defines the technology used and how it fits together and names the objects.
This should reference the source and destination definitions
Implementation standards
This references the standards to be followed and may include templates or examples to be followed.
If the objects are generated from the definitions it will name the generating objects and how they are to be used to release the final object.
Each ETL object should centrally record information about steps carried out – this may be as detailed as logging each individual statement for slower batch systems or could be more high level. For systems that work in very small increments this may need to be configurable.

## Testing

The result needs to be tested against the source data and will depend very much on what is available.
At least data elements of entities should be reconciled preferably automatically.
Any totals across entity classes should also be checked. This may depend on the source system from which the data is extracted but where data for an entity is extracted from multiple systems care should be taken to check totals from each system to avoid duplications. This check should be automated to prevent errors creeping in when changes are made.

## Scheduling

Definition of what needs to be scheduled and how.
This is best if there is a single scheduling control implementation for the system but this may not be possible.
This will usually schedule the ETL objects and includes periodicity (time based/polling) and time windows as well as any dependencies.
The format of the definition needs to fit in with the scheduling implementation.
The scheduler should record what is scheduled and the start/end times.

## Troubleshooting/Alerting

An alerting method needs to be implemented. This could be as simple as someone checking a status periodically or fit in with a corporate alerting mechanism to alert a dedicated team.
This describes action to be taken in the event of a failure. What information will be available and how to restart the system and will be closely allied with the scheduling implementation.
It will describe how to track back to the source of an error usually starting with information provided by the scheduler.
For the implemented system data checks should be made as part of the scheduled run. Any errors identified should be included in the check so that the maintenance team gets early indication of issues without waiting for the users.
All ETL and scheduling objects should provide monitoring information – record counts, start/end times and this should be checked and an alert issued. This data should be secured so that any issues (particularly with performance) can be retrospectively identified.