

Standard Business Reporting

Australian Taxation Office

ATO XML Schema Generation Algorithm

Date: 19 June 2019

Status: Final

* This document and its attachments are **Unclassified.**

 For further information or questions, contact the SBR Service Desk at [SBRServiceDesk@sbr.gov.au](http://sbr.gov.au/agreement/Gateway/1.0/Push/PKI) or call 1300 488 231.

International callers may use +61-2-6216 5577

VERSION CONTROL

| **Version** | **Date** | **Description of changes** |
| --- | --- | --- |
| 1.2 | 19/06/2019 | Updated with functional changes as described in new section:Section 2.7.10 - Other Tuple scenarios for shortening element namesThe functional changes will impact generation of some schemas only.The functional changes will not take affect until the next versioning of affected services. |

***Note:*** *Previous Version Control is located at Appendix A of this document.*

ENDORSEMENT

APPROVAL

 Chief Solutions Architect

 Standard Business Reporting

Michael Ferris Project Manager

 Digital Wholesale Services

 Australian Taxation Office

Copyright

© Commonwealth of Australia 2017 (see exceptions below).
This work is copyright. Use of this Information and Material is subject to the terms and conditions in the “SBR Disclaimer and Conditions of Use” which is available at [http://www.sbr.gov.au](http://www.sbr.gov.au/software-developers/enabling-sbr-in-my-application/sbr-taxonomy/view-taxonomy). You must ensure that you comply with those terms and conditions. In particular, those terms and conditions include disclaimers and limitations on the liability of the Commonwealth and an indemnity from you to the Commonwealth and its personnel, the SBR Agencies and their personnel.

You must include this copyright notice in all copies of this Information and Material which you create.  If you modify, adapt or prepare derivative works of the Information and Material, the notice must still be included but you must add your own copyright statement to your modification, adaptation or derivative work which makes clear the nature of your modification, adaptation or derivative work and you must include an acknowledgement that the adaptation, modification or derivative work is based on Commonwealth or SBR Agency owned Information and Material. Copyright in SBR Agency specific aspects of the SBR Reporting Taxonomy is owned by the relevant SBR Agency.

Table of contents

[1 Introduction 5](#_Toc498687687)

[1.1 Purpose 5](#_Toc498687688)

[1.2 Audience 5](#_Toc498687689)

[2 ATO XML Schema generation Algorithm 6](#_Toc498687690)

[2.1 Background 6](#_Toc498687691)

[2.2 Relationship with the SBR Definitional Taxonomy 6](#_Toc498687692)

[2.3 Single namespace 6](#_Toc498687693)

[2.4 Schema facets 7](#_Toc498687694)

[2.5 Reusing Schemas for multiple actions 8](#_Toc498687695)

[2.6 Context Structure Table 9](#_Toc498687696)

[2.7 Message Structure Table 9](#_Toc498687697)

[3 Naming conventions and files 18](#_Toc498687698)

[3.1 ATO Schema publication 18](#_Toc498687699)

[3.2 ATO generated schema naming convention 18](#_Toc498687700)

[4 Single Touch Payroll (STP) Example 20](#_Toc498687701)

[4.1 Pay Event 20](#_Toc498687702)

[4.2 Pay Event Child 21](#_Toc498687703)

# **Introduction**

Purpose

The purpose of this document is to describe the algorithm used to generate XML Schemas from the Message Structure Table (MST) spreadsheet/s. The algorithm will be applied to services produced from May 2017 onwards.

The ATO Common Message Implementation Guide (cMIG) currently provides a description of the information contained within an MST and therefore should be used as a reference to this document.

Audience

The audience for this document is any organisation that will be building ATO SBR services that use XML rather than XBRL. Typically this will be software application developers.

# **ATO XML Schema generation Algorithm**

* 1. Background

The ATO has to date, predominantly built services for the SBR program using XBRL as the message payload format. In recent times there has been a move to building services in XML[[1]](#footnote-2).

An initial algorithm was introduced in 2014 to allow XML schemas to be generated based on the XBRL design artefacts. This design decision was made at a time when it was anticipated that the ATO was likely to produce the same report/message contract exchange in multiple formats. The ATO is only expecting to use multiple formats for a service action in relatively rare cases.

While functional to a point, the original algorithm still ties message design to XBRL principles and has proven to be overly complex and confusing – especially for new software developers who have not been on the XBRL journey. The algorithm also produces message sizes that are too large.

After consultation via the Software Developer Working Group (TWG), the ATO has proceeded to improve the algorithm used to generate XML schemas.

* 1. Relationship with the SBR Definitional Taxonomy

The ATO is still committed to the SBR vision of reducing complexity through having one data dictionary for the whole of government. As such, SBR definitional terms will be used as the basis for ALL message formats including XML.

Changes to the ‘over-the-wire’ format are now allowed by SBR principles. The ATO is taking an algorithmic approach to changing the ‘over-the-wire’ format for XML. Alternatively making manual changes to SBR Element Names would introduce an XML taxonomy, which would need to be managed separately.

* 1. Single namespace

To simplify the complicated XBRL namespaces/taxonomy versioning principles, versioning has been removed from namespaces. The root element of the XML message is based upon the report’s acronym and optionally the service action. This is the same approach used to resolve the versioning irritant for XBRL reports that use the new taxonomy architecture.

The namespace in an XML message is defined as [http://www.sbr.gov.au/ato/{contractname}](http://www.sbr.gov.au/ato/%7Bcontractname%7D)

As an example, the Pay Event namespace in an XML message is defined as [http://www.sbr.gov.au/ato/payevent](http://www.sbr.gov.au/ato/ABNREGREQUEST%20) .

The use of a single namespace for a report enables the replacement of format rules with schema facets.

This does have an impact on third parties as described below.

| **Impact** | **Recommendation** |
| --- | --- |
| Reused elements such as AddressDetails no longer have a common namespace across multiple messages.  | Use shaping/interfaces to minimize the impact on your software. |
| Namespaces do not change over multiple years. | In some case the same schema can be re-used over multiple years even where there are some business element or business rule changes.For example if there are new optional data element(s)\features in this year’s schema that your customers do not use then:* either use last year’s schema OR
* use this year’s schema

i.e. you do not have to use both of them concurrently.If you need to support more than one version of a schema concurrently – then use the features of your programming language to separate the two versions. For example, in .NET this would be done by applying separate .NET namespaces.Use shaping/interfaces to minimize the impact on your software. |

* 1. Schema facets

Schema facets will be used in preference to business rules wherever possible. Schema Facets allow for data restrictions that extend past simple data type checks and can typically replace mandatory and monetary format business rules.

Elements in the SBR AU Taxonomy are defined broadly for use by all government agencies in the SBR program. This broad usage has meant that the ATO has historically had to use format rules to cater for report specific restrictions. Monetary data elements are a typical example where there are no length restrictions in the SBR definitional data elements but the ATO needs to restrict data element length.

The XML algorithm and the use of a report specific namespace allows the ATO to start replacing format business rules with schema facets.

Schema facets that the ATO will support are shown in the following table. Note that there are XML schema restrictions on how these can be applied. It is beyond the scope of this document to fully describe how schema facets work. See <http://www.w3.org/TR/xmlschema-2/> for further technical information.

| **MST Column** | **Schema Facet** | **Usage** |
| --- | --- | --- |
| Min Occurs | minOccurs | Minimum number of times an element should be provided. 1 is used for mandatory elements. |
| Max Occurs | maxOccurs | Maximum number of times an element can be provided. Will have a value for 1 except for repeating elements.  |
| Min Length | minLength | Specifies the minimum number of characters or list items allowed. |
| Max Length | maxLength | Specifies the maximum number of characters or list items allowed.  |
| Total Digits | totalDigits | Specifies the exact number of digits allowed.  |
| Fractional Digits | fractionDigits | Specifies the maximum number of decimal places allowed.  |
| Min Inclusive | minInclusive | Specifies the lower bounds for numeric values (the value must be greater than or equal to this value) |
| Max Inclusive | maxInclusive | Specifies the upper bounds for numeric values (the value must be less than or equal to this value) |
| Pattern | pattern | Defines the exact sequence of characters that are acceptable |
| Full Enumeration | enumeration | Defines a list of acceptable values |

* 1. Reusing Schemas for multiple actions

Where a business entity is used for multiple actions then the same schema will be used where possible. For example, in Activity Statements (AS), if this approach was used the same AS schema would have been used for:

* The lodge request and response.
* The prelodge request and response.
* The prefill response.

When a schema is re-used in this way or is an expected future change – then the service action will not be included in the main schema element name or the namespace.

It should be noted that when a schema is re-used over multiple actions there will be a subsequent limitation on how many schema facets can be included into the schema. The lowest common denominator across multiple actions will need to be used in these cases.

* This is most likely to affect minimum occurrence restrictions.
* If an element is optional for one action, then it will end up being optional for all of them.
* This will result in business rules being used instead of minimum occurrence restrictions in the schema.

### Impact of Prefill on Schema Reuse

The ATO will usually have less data that it can return in a prefill than is required to be submitted through a lodgment action. When the same schema is to be used for prefill and lodge (or another action) then the ATO is faced with three primary choices:

|  |  |  |
| --- | --- | --- |
| **#** | **Choice** | **Implication** |
| 1 | Treat the prefill as the lowest common denominator for schema facets. | This will result in many data fields having their Minimum Occurrence changed to be optional. Business rules will have to be used instead. There will also be cases where data previously submitted via another channel (e.g. paper) will not conform to the format the ATO actually wants the data in. This could result in other schema facets being removed and replaced with business rules.This provides for re-use but is a higher cost option. |
| 2 | Do not treat prefill as the lowest common denominator for schema facets. | In this case the prefill is not used as the lowest common denominator for schema facets. This implies that the ATO prefill messages will not necessarily conform to the XML schema. Third parties would need to take this into account – and certainly not try to run schema validation over ATO prefill responses.This provides for re-use and is the lowest cost option.  |
| 3 | Use a separate schema for prefill. | Having a separate schema also means that the prefill will not be used as the lowest common denominator.This option hampers re-use (shaping and interfaces can still be used) and is the highest cost option. |

The appropriate choice will be made through consultation with third party software developers via the TWG.

* 1. Context Structure Table

The Context Structure Table (CST) will not be used for XML based designs. Its usage for XML schema generation will be phased out in two stages in order to reduce risk.

In phase 1, use of the CST will be dramatically reduced and in phase 2 it will no longer be used for XML based services.

*After phase 1 has been completed – this section will be revisited.*

### Phase 1

In phase 1 the CST will still exist, though usage of contexts will be constrained to tuple based principles.

Any context defined in the CST will become a top level XML element in the schema.

* Tuple naming standards will be used.
* Most CST data elements will be marked as N/A. This includes:
	+ Start/Instant Date
	+ End Date
	+ Period Type
	+ Identifier Scheme
	+ Identifier Value
* Dimension details will be left blank.
* The only columns that will be used are those that are similarly used for tuples in the Message Structure Table (MST).
	+ Seq Number
	+ Label
	+ Description
	+ Min
	+ Max

There is a limitation in phase 1 in that it will not be possible to include simple data elements (Facts) at the root level of a report schema. Facts always exist under tuples that are derived from the Context names (or tuples in the MST).

### Phase 2

In phase 2, support for the CST will be removed for XML services. All message design will be represented in the MST.

**Moving from Phase 1 to Phase 2**

Moving from phase 1 to phase 2 will not be breaking with regards the XML schema generated. Any contexts created in phase 1 will be defined as a top level tuple in the MST for phase 2.

The Pay Event service (part of the Single Touch Payroll initiative) is the only service going through Phase 1 and 2.

* 1. Message Structure Table

The Message Structure Table (MST) describes the message payload in a format that can be understood by BA’s, developers and testers. This artefact will always show the path back to the SBR definitional taxonomy, regardless of message payload format.

This section outlines the relationship between the MST and the resulting XML schema, with the primary purpose being to describe the algorithm that is used to reduce the length of SBR data element names.

The essence of the techniques described in this section take advantage of the fact that SBR data elements are defined in the format <ObjectClass>{.<ObjectClass>}.Property.Classword.

* ObjectClass names are removed from element names through the use of Tuples.
* Property names that can be considered to start with ObjectClass names are also reduced in size using Tuples.
* The Classword is shortened.

As the MST will still describe the message in terms of SBR data elements, the algorithm for generating an XML Schema must be:

* Simple enough for most people to understand. This ensures that the relationship between the MST and the XML Schema is relatively intuitive.
* Consistently applied across all data elements rather than mixing and matching in ’special circumstances’.

### MST Alignment

The following table highlights some of the key alignments between the MST and the associated XML schema.

| **Algorithm** | **Detail** |
| --- | --- |
| Tuple | Tuple will still be used as the Form Element Type.  |
| Ordering | Schema data elements are aligned with the MST order. |
| Cardinality | In a XML message, cardinality of all elements and tuples will be enforced by schema validation. The only exception to this rule will be when an MST is used over multiple service actions – in this case business rules may be used to cater for the possibility that an element is mandatory for one action but optional for another.The ‘Min’ & ‘Max’ columns with the MST will be used to define cardinality for all elements/tuples.All elements in the XML schema use the ‘minOccurs’ attribute to indicate a value of ‘0’ or ‘1’. The ‘maxOccurs’ attribute is used only when an element repeats (i.e. maxOccurs > 1).  |
| All data type columns | The data type columns align to schema facets.* Data Type
* Pattern
* Full Enumeration
* Min Length
* Max Length
* Total Digits
* Fractional Digits
* \* Min Inclusive
* \* Max Inclusive

\* These are yet to be approved via the TWG. |
| Guidance | Guidance is placed in the XML Schemas as annotation documentation elements. |

### MST Differences

The following table highlights the key differences between the MST and the associated XML Schema.

|  |  |
| --- | --- |
| **Algorithm** | **Detail** |
| Headings | Headings are used in MSTs to assist readability. As per XBRL, headings in a MST play no part in the physical structure of an XML message/schema. |
| Namespace Prefix | Unlike XBRL, elements within XML messages are defined within a single message schema and do not retain the various namespace prefixes with versions that XBRL element have.The namespace prefix refers back to the version of an element that was used as a basis for the XML design. As for the Element Names (see below) this shows the path back to the SBR definitional element. For example pyde.02.00 may be shown as the namespace prefix for AddressDetails.Line1.Text. |
| Element Names | Element Names are specified using the full SBR definitional data element name. This is intentional as it shows the path back to the SBR definitional element.In order to meet some of the guiding principles, element names in XML messages will be slightly different to the MST element names. The following is applied in addition to the ObjectClass, Property and Classword shortening techniques described later in this document. * Full Stops are removed from element names
* The hyphen character (‘-’) is changed to ‘Minus’ in both context acronyms and element names.
* Numeric digits at the start of a name are removed
* Any name that has characters in the following regular expression are removed [\s\r\n\{\}\.:'\(\),/]\*.

***Note:*** *some of the logic is purely defensive in nature (e.g. removing whitespace) as this should not occur for properly defined tuple and element names.* |
| Unused columns | The following MST columns are not relevant to XML services:* Context Instance (including Explicit Tuple Conditions)
* Period Type (will be shown as a reference but will not affect the structure).
 |
| No Context Facts | Context Facts are an XBRL only concept and therefore only Facts will be used for XML. |

### Classword shortening

The last portion of SBR data element names is referred to as the Classword. The classword is used to indicate the type of data that is being used. While ideally this could be removed entirely, in too many cases this results in a name collision and in too many cases removal would make the meaning of an element too obscure.

|  |  |
| --- | --- |
| **Current element postfix** | **Replaced with** |
| Amount | A |
| Code | C |
| Count | Ct |
| Date | D |
| Datetime | Dt |
| DayofMonth | Dm |
| Description | De |
| Fraction | F |
| Identifier | Id |
| Indicator | I |
| Month | M |
| Number | N |
| Percent | P |
| Rate | R |
| Text | T |
| Timestamp | Ts |
| Type | Ty |
| Year | Y |
| <anything else> | Extracted capitalsThis is a fall-back position that is unlikely ever to be utilised. This should be considered as a defensive programming mechanism. |

### Removed ObjectClass names

The following ObjectClass names provide relatively little business value and will be removed entirely.

|  |  |
| --- | --- |
| **Object Class Name** | **Reasoning** |
| Identifiers | Already captured by replacement of Identifier postfix to Id.E.g. Identifiers.AustralianBusinessNumber.Identifer becomes AustralianBusinessNumberId |
| Report | This prefix does not add any value.e.g. Report.AmendmentReason.Text becomes AmendmentReasonT |

### Using Tuples to remove ObjectClass Names

For each ObjectClass name used in an element name:

* If there is a direct match between an ObjectClass name and any parent tuple name, then the ObjectClass name is removed from the data element name.
* If any parent tuple name contains the ObjectClass name, then the ObjectClass name is removed from the data element name.

#### Example 1 - Direct match

In this example both Capital and ThinCapitalisation are removed from the element name.

|  |  |
| --- | --- |
| **MST Path** | Capital ThinCapitalisation |
| **SBR Element Name** | Capital.ThinCapitalisation.AssetRevaluation.Amount |
| **XML Element Name** | AssetRevaluationA |

#### Example 2 - Partial match for Thin Capitalisation

In this example both Capital and ThinCapitalisation are removed from the element name using a slightly flatter hierarchy.

|  |  |
| --- | --- |
|  **MST Path** | CapitalThinCapitalisation |
| **SBR Element Name** | Capital.ThinCapitalisation.AssetRevaluation.Amount |
| **XML Element Name** | AssetRevaluationA |

#### Example 3 - Partial match for AddressDetails

In this example AddressDetails is removed from the element name as there is a partial match with the AddressDetailsPostal tuple name.

|  |  |
| --- | --- |
| **MST Path** | AddressDetailsPostal |
| **SBR Element Name** | AddressDetails.Line1.Text |
| **XML Element Name** | Line1T |

### Using Tuples to remove start of ObjectClass and Property Names

Some ObjectClass and Property names start with a word that can also be considered to be an ObjectClass. In order to further reduce the data element names, the following is done:

* If the ObjectClass or Property name starts with the name of any parent tuple then that word will be removed from the ObjectClass\Property name.

Note that “words” are based on the capitalisation used within ObjectClass\Property names. So OrganisationalName is considered to contain the two words Organisational and Name.

#### Example 1 – Starts with name of parent tuple

In this example IncomeTax is removed as a direct match but AttributionManagedInvestmentTrustClass and AttributionManagedInvestmentTrustMembers have their names shortened as they start with AttributionManagedInvestmentTrust.

|  |  |
| --- | --- |
| **MST Path** | IncomeTax AttributionManagedInvestmentTrust |
| **SBR Element Names** | IncomeTax.AttributionManagedInvestmentTrustClass.DescriptionIncomeTax.AttributionManagedInvestmentTrustMembers.Count |
| **XML Element Names** | ClassDeMembersCt |

#### Example 2 – Starts with a word used in name of parent tuple

In this example Organisational is a considered to start with the word Organisation. Just removing Organisation would leave a trailing “al” in the element name. Removing the whole word means that the trailing “al” is also removed.

|  |  |
| --- | --- |
| **MST Path** | Organisation |
| **SBR Element Name** | OrganisationNameDetails.OrganisationalName.Text |
| **XML Element Name** | NameDetailsNameT |

### Applying Usage and Currency to common tuples

Usage (Postal, Business etc.) and Currency (Current vs Previous) will be applied using naming standards rather than via special XBRL Context names or Explicit Tuples (see the cMIG for details on Explicit Tuples).

This naming standard will be applied to the end of tuples. While it could equally be placed at the start of element names – using the end tends to group like tuples. This is a principle of showing the most important part of the data element first rather than last. (AddressDetails being more important than Usage or Currency). The following list shows the mapping between commonly used Explicit Tuple condition values and the name that will instead be applied to the end of tuple names.

* C’ -> Current
* ‘P’ -> Previous
* ‘POS’ -> Postal
* ‘BUS’ -> Business
* ‘RES’ -> Residential
* ‘LGL’ -> Legal
* ‘01’ -> Personal (Electronic Contact Telephone, Facsimile, Mail usage)
* ‘02’ -> Business (Electronic Contact Telephone, Facsimile, Mail usage)
* ‘03’ -> Contact (Electronic Contact Telephone, Facsimile, Mail usage
* All others will use the current code value e.g. TrueAndCorrect for declaration.

This mechanism will be applied to the following tuples:

* AddressDetails
* ElectronicContactTelephone
* ElectronicContactElectronicMail
* ElectronicContactFacsimile
* Declaration
* PersonNameDetails
* PersonUnstructuredName
* FinancialInstitutionAccount
* OrganisationNameDetails

Due to this change the following data elements will NOT be used in XML schemas:

* AddressDetails.Currency.Code
* AddressDetails.Usage.Code
* ElectronicContact.ElectronicMail.Usage.Code
* ElectronicContact.Facsimile.Usage.Code
* ElectronicContact.Telephone.Usage.Code
* OrganisationNameDetails.Currency.Code
* OrganisationNameDetails.OrganisationalNameType.Code
* PersonNameDetails.Currency.Code
* PersonNameDetails.Usage.Code
* PersonUnstructuredName.Usage.Code
* Declaration.StatementType.Code

### Creating a single schema element for matching Tuples

To enable re-use of schema elements, matching tuples within the MST will be combined into a single schema element. This will typically be the case for tuples such as AddressDetails.

A re-useable schema element will be created when:

* The Taxonomy Element Names all match and are in the same order.
* All data element types, min occurs, max occurs, maximum length also match.

Note that while re-using schema elements does provide some benefits the following drawbacks will be encountered:

* Some schema facets may have to be replaced with business rules – as they may not apply to all instances of the tuples.
* Guidance is currently created based on the first tuple in the MST so FAID specific guidance will be lost.

### Tuples with a single data element

In some cases a tuple will be used in the MST just to provide a different business meaning to a single data element.

For example:

|  |  |  |
| --- | --- | --- |
| **Tuple Path** | **SBR Element Name** | **XML Element Name** |
| Primary | Income.Net.Amount | PrimaryIncomeNetA |
| NonPrimary | Income.Net.Amount | NonPrimaryIncomeNetA |

In this case the business meaning of Income.Net.Amount is being adjusted by the parent tuple so that it caters for Primary Income and Non Primary Income.

The algorithm recognises that the purpose of the tuple is just to provide a different business meaning and in the resulting XML schema the tuple and data element will be combined into a single data element.

So instead of directly reflecting the MST structure of:

 Primary

 IncomeNetA

 NonPrimary

 IncomeNetA

The Xml Schema will have:

 PrimaryIncomeNetA

NonPrimaryIncomeNetA

While an alternate approach would be to directly modify the SBR Element Name with a new ObjectClass name, the ATO has chosen not to do so as the path back to the original SBR Element Name would be lost.

### Other Tuple scenarios for shortening element names

In some cases, the algorithm modifies element names based on a combination of the name shortening algorithms described in Sect 2.75 and 2.7.9 as per example below,

|  |  |
| --- | --- |
| **MST Path** | SuperannuationContribution EmployerContributions  PersonalContributions |
| **SBR Element Name (Single element within parent tuple)** | SuperannuationContribution.PersonalContributions.Amount |
| **XML Element Name** | PersonalContributionsA |

The shortening algorithm in Section 2.7.5 is first applied and produces a shortened name of “A”.

Subsequently the algorithm described in Section 2.7.9 is applied to produce a final shortened name of “PersonalContributionsA”.

### Unsupported features

The following features have been considered but are not supported by the algorithm.

| **UnsupportedFeature** | **Reasoning** |
| --- | --- |
| Lookup reference to change element names | One relatively easy way of reducing the length of data element names is to provide a lookup reference that matches an SBR data element name to a shortened name to be used for XML.While a lookup based implementation has been considered, this had not been done as it is too likely that this would be abused. Consistency in element naming is considered more important than element length. |
| Manually designed schemas | The algorithm will result in an XML Schema that does not align with how a person would manually design a schema. The resulting schema will still be useable for the intended consumer groups – developers and testers. |
| Heuristic name matching | There are algorithms that can provide a heuristic match on element names. Using such an algorithm has not been considered as matching ObjectClass names against Tuple Names is more conformant with SBR naming principles. This has not been considered as it will make the relationship between the MST and the XML Schema less intuitive.  |
| Stopping the removal of all but the Classword  | It is possible that the use of Tuple names to shorten SBR element names can lead to only the shortened classword being left as the XML element name.For example:Parent Tuples: Income Superannuation LumpSumPayment.SBR Element Names:Income.Superannuation.LumpSumPayment.DateIncome.Superannuation.LumpSumPaymentTaxableComponentTaxedElement.AmountXML Element Names:DTaxableComponentTaxedElementAWhile D as an element name is not meaningful – it is expected that developers will understand this without any difficulty. In addition, the schema will be annotated with documentation that describes the element to assist in that understanding. It is not expected that this will happen very often. |
| Automatically converting ObjectClass to tuples | It is technically feasible to automatically create Tuples (XML Elements) from the ObjectClass’s used in data element names. This has not been considered as it will make the relationship between the MST and the XML Schema less intuitive.  |
| Shortening single data elements | Occasionally the algorithm will not shorten a data element name as there is no match between the data element’s ObjectClass names and the data element’s parent tuple names. While these elements will not have their name shortened it is considered a good thing that they get their full meaning shown through a longer element name.Consider the following example:Parent Tuple Name: PayrollSBR Element Name: PaymentRecord.Transaction.DateXML Element Name: PaymentRecordTransactionDWhile the classword has been reduced in size – the XML name is otherwise relatively long.  |

# **Naming conventions and files**

The XML schema naming convention is described in the ATO Taxonomy Architecture and Naming Conventions document. See <http://www.sbr.gov.au/software-developers/developer-tools/ato> for details. **This section is not the source of truth and will be removed at a later date**.

* 1. ATO Schema publication

ATO generated schemas will be placed in the Service Suite zips along with service artefacts. They will not be published on YETI or on SBR ShareFile.

* 1. ATO generated schema naming convention

### New conventions at a glance

The ATO names report schemas as:

***ato.[message acronym].[collaboration].[year].[action].[response].[major version].[minor version].xsd***

* The acronym of ‘ato’ has been added.
* A new component of ‘year’ has been added.
* The component of ‘action’ becomes optional so a schema can be re-used over multiple actions if necessary.
* The component of ‘request’ has been omitted.
* The component of ‘response’ will be used when the response is different to the request schema.

### In detail

| **Name Component** | **Description and value** | **Optionality** | **Comments** |
| --- | --- | --- | --- |
| ato | Agency acronym identifier* *set to ‘ato’*
 | Mandatory | * The ‘ato’ prefix has been added for clarity and to negate any potential clashes in the future at a whole of government level with other agencies.
 |
| message acronym | Message acronym identifier* *set to message acronym [value]*
 | Mandatory | * Relationships of messages for ‘parent’ and ‘child’ are found in the Service Registry.
 |
| collaboration | Collaboration numeric* *set to numeric [nnnn]*
 | Mandatory | * Collaboration numeri are defined at the message level, with the ‘parent’ message typically being the service.
 |
| year | Year in the which the message is to be deployed to production* *set to year [ccyy]*
 | Mandatory | * A message could be reused over a number years with a service being versioned yet not the message itself.
 |
| action | Action descriptor for a message used in a service* *set to an action as defined in platform specific WIGs [value]*
 | Optional | * Where a message is to be re-used across multiple actions this name component would be absent.
* An action over multiple planforms is first defined by SBR Core Services WIG (a pre-lodge would be in the schema instead of validate as an example).
 |
| response | Outbound response message for a request action* *set to ‘response’*
 | Optional | * The deletion of the ‘request’ component to the name implies the message is both a request and response – thus allowing for re-use.
* The change to an optional ‘response’ component in the name would only be necessary if a response message schema is different to the request message schema.
 |
| major version | Major version numeric* *set to numeric [nn]*
 | Mandatory | * Set to ‘01’ for first release.
* Incremented where there is a break to the interface.
 |
| minor version | Minor version numeric* *set to numeric [nn]*
 | Mandatory | * Set to ‘00’ for first release.
* Incremented where there is a minor interface change.
 |

### Comparative examples

| **File** | **Pre January 2017 conventions** | **January 2017 onward conventions** |
| --- | --- | --- |
| XML | abnregtaxadd.0001.lodge.request.01.00.xsd | ato.abnregtaxadd.0001.2016.lodge.01.00.xsd |
|  | abnregent.0001.prefill.request.01.00.xsd  | ato.abnregent.0001.2016.prefill.01.00.xsd |
|  | abnregent.0001.prefill.response.01.00.xsd | ato.abnregent.0001.2016.prefill.response.01.00.xsd |

***ATO Disclaimer:*** *The examples above show the schema names if the messages were done in January 2017.They are in no way intended to imply the ATO is changing these services.*

# **Single Touch Payroll (STP) Example**

The following example is for the STP program of work and is not an official representation as the actual message structure tables for STP may be subject to further changes.

Rather than show the schema or an XML example – the element names will be shown with tabs being used for the hierarchy. Where a tuple is based on a context defined in the CST during phase 1 “(c)” will be placed next to the name.

* 1. Pay Event

Rp (c)

SoftwareInformationBusinessManagementSystemId

AustralianBusinessNumberId

WithholdingPayerNumberId

Organisation

DetailsBranchC

NameDetailsNameT

PersonUnstructuredNameFullNameT

ElectronicContact

ElectronicMailAddressT

TelephoneMinimalN

AddressDetailsPostal

 Line1T

 Line2T

StateOrTerritoryC

PostcodeT

CountryC

Payroll

PaymentRecordTransactionD

LabourStatisticsEmployeesCt

MessageTimestampGenerationDt

IncomeTaxPayAsYouGoWithholdingTaxWithheldA

Remuneration

PayrollTransactionId

WagesAndSalariesPlusPaymentsOtherA

FullFileReplacement

AmendmentIndicator

RemunerationTransactionId

Declaration

SignatoryIdentifierT

SignatureD

StatementAcceptedI

Int (c)

AustralianBusinessNumberId

TaxAgentNumberId

ElectronicContact

ElectronicMailAddressT

TelephoneMinimalN

PersonUnstructuredNameFullNameT

Declaration

SignatoryIdentifierT

SignatureD

StatementAcceptedI

* 1. Pay Event Child

In addition to using (C) to indicate a context has been used in phase 1 – (t) will be used to indicate that a tuple has been placed into the MST to modify the name of a single field.

Payee (c)

EmploymentPayrollNumberId

PreviousEmploymentPayrollNumberId (t)

TaxFileNumberId

AustralianBusinessNumberId

PersonNameDetails

FamilyNameT

GivenNameT

OtherGivenNameT

PersonDemographicDetailsBirthD

EmployerConditionsEmployment

StartD

EndD

AddressDetails

 Line1T

 Line2T

StateOrTerritoryC

PostcodeT

CountryC

ElectronicContact

ElectronicMailAddressT

TelephoneMinimalN

AllowanceItem (note change to AllowanceRenumerationAndIncomeTax to shrink the elements below)

IncomeTaxOtherAllowanceTypeD

Remuneration

AllowanceTypeC

IndividualNonBusinessEmploymentAllowancesA

DeductionItem

Remuneration

DeductionTypeC

DeductionA

EmployerSuperannuationContribution

ContributionsSuperannuationGuaranteeA

ReportableA

RemunerationOrdinaryTimeEarningsA

WageItem

Remuneration

UnusedAnnualOrLongServiceLeavePayment

LumpSumAC

LumpSumAA

LumpSumBA

LumpSumDA

LumpSumEA

FringeBenefitsReportable

TaxableIncomeA (t)

ExemptIncomeA (t)

OtherSpecifiedRenumerationAndIncomeTax

SpecifiedByRegulationPaymentGrossA

PayAsYouGoWithholdingTaxWithheldA

VoluntaryAgreementRenumerationAndIncomeTax

VoluntaryAgreementGrossA

PayAsYouGoWithholdingTaxWithheldA

JointPetroleumDevelopmentAreaRenumerationAndIncomeTax

JointPetroleumDevelopmentAreaPaymentA

ForeignWithholdingA

PayAsYouGoWithholdingTaxWithheldA

WageItem (continued – under Payee)

WorkingHolidayMakerRenumerationAndIncomeTax

WorkingHolidayMakerGrossA

PayAsYouGoWithholdingTaxWithheldA

IndividualNonBusinessRenumerationAndIncomeTax

IndividualNonBusiness

GrossA

CommunityDevelopmentEmploymentProjectA

ExemptForeignEmploymentIncomeA

PayAsYouGoWithholdingTaxWithheldA

BusinessAndPersonalServicesRenumerationAndIncomeTax

LabourHireArrangementPaymentGrossA

PayAsYouGoWithholdingTaxWithheldA

ForeignEmploymentRemunerationAndIncomeSummary

InternationalDealingsTaxOffsetA

TaxPayAsYouGoWithholdingTaxWithheldA

PaymentToForeignResidentGrossA

TerminationRenumerationAndIncomeTax

PayAsYouGoWithholding

EmploymentTerminationPaymentTypeC

TaxWithheldA

Superannuation

TaxFreeComponentA

EmploymentTerminationTaxableComponentTotalA

PaymentRecordPaymentEffectiveD

Onboarding

PaymentArrangementPaymentBasisC

ResidencyTaxPurposesPersonStatusC

TaxOffsetClaim

TaxFreeThresholdI

SeniorAustraliansTaxOffsetI

ZoneI

OverseasForcesI

ParentSpouseOrInvalidRelativeI

DownwardVariationA

StudentLoan

HigherEducationLoanProgrammeI

TradeSupportI

StudentFinancialSupplementSchemeI

Declaration

SignatoryIdentifierT

SignatureD

StatementAcceptedI

# **Appendix A – Previous Version Control**

| **Version** | **Date** | **Description of changes** |
| --- | --- | --- |
| 1.1 | 23/11/2017 | Updated with the following **non-functional** change:**Deleted :****‘Section 1.3 RELATIONSHIP OF THIS DOCUMENT TO THE ATO COMMON MIG**The intention is to merge the detail within this document into the ATO Common Message Implementation Guide (cMIG) at a later date.’.*The cMIG is being deconstructed into smaller discrete reference documents and is expected to be fully replaced 14th December 2017.* |
| 1.0 | 31/07/2017 | Versioned to Final. No functional change. |
| 0.1 | 27/04/2017 | First Draft. |

1. XML Specification: [https://www.w3.org/TR/XML/](https://www.w3.org/TR/xml/) [↑](#footnote-ref-2)