Overview

The Technical Document *Framework* (TDF) provides the configuration tools and authoring User Interface for the Technical Documentation *Application* (Tech Docs). It provides the core functionality to enable the creation of different types of Document content. As compared to documents that store data in individual files, ‘Document’ in this context refers to a collection of text and graphic (image) data, that conforms to a well-defined document *structure*, which is stored as XML with an associated Item. This content can then be *published* into one or more files in a file system, but that is not its native storage format.

The Technical Documentation Application is a content authoring environment, fully integrated into Aras Innovator PLM that collectively provides component content management capability for topics-based, modular documentation. Tech Docs enables users of various disciplines to author, visualize, share, and publish information all within a central, controlled, and collaborative environment. Unlike other systems that provide Document-level Management functions for content produced by external applications, Aras Tech Docs leverages Innovator’s core PLM functionality (Access Control, Workflow/Lifecycle, Change Notification, etc.) and provides a web-based authoring tool for creating content directly, embedding shared document components, or referencing other elements managed by the PLM system. So as related Documents, Parts, Project Tasks, Requirements, or any other information managed by Aras Innovator evolve through subsequent revisions, authors of technical documentation can be alerted and navigated to pending changes, or the system can update the related documentation automatically.

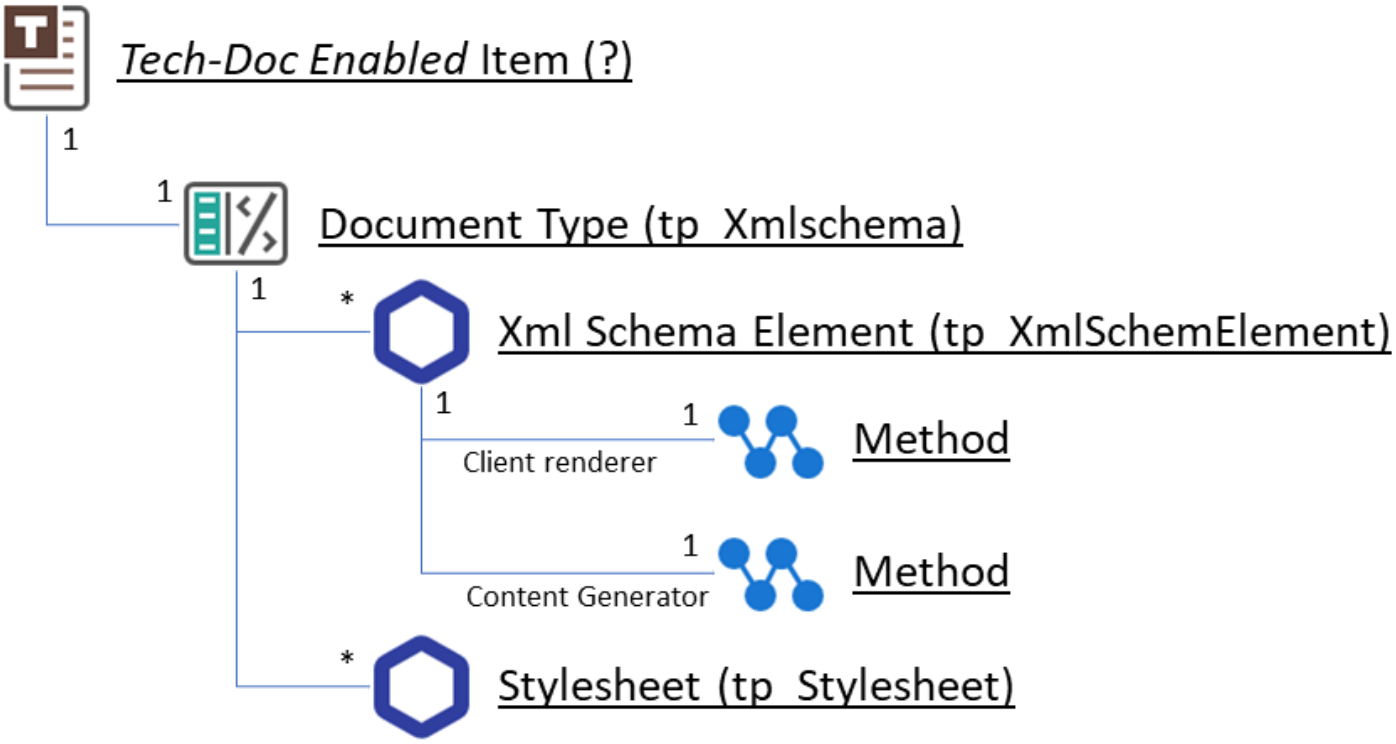
Technical Documents can be composed of individual components (sections, paragraphs, illustrations, tables, etc.), where each component can change independently; and optionally be created/managed by multiple users. The aggregated information forms the complete published document. This Structured Content is specified by a user-defined data model that identifies the elements of each component, their relationships and cardinality, and any restrictions on data type and value limitations. The built-in authoring User Interface directs the author as to the types of document elements that can be inserted, constantly validating the document as it is created. Candidate documents for release can be routed through workflow and published on final acceptance.

This document describes the functions and features of the Technical Documentation Framework, which is used to configure Technical Documentation. It is meant to be a reference for the Technical Document Administrator for configuring technical content.

- *Overview* provides a brief explanation of the major components of a Technical Document and the Technical Document Framework.
- *Document Type Configuration* describes how to configure a Document Type.
- *Customizing Technical Documents* provides a brief overview of configuring the **tp_Block** ItemType, which is the Item used for the Technical Documentation Application
- *Content Generation* describes Content Generators and how to create them.
- *Item Reference Configuration* discusses Item Reference Configuration and how/when to use them.
- *Document Filters* shows how to configure the Filters/Options for a Technical Document
- *Appendix* includes subsections describing additional useful information for configuring a Technical Document and customizing the environment.

Anatomy of a Technical Document

In its most basic instance configuration, a Technical Document-enabled Item is a single entity that contains formatted text, stored as XML in a single Property, and one or more associated graphics. Each of these Items references a Document Type that specifies the XML Schema, CSS Styling, and any configured content renderers/generators for automating the creation of XML content.



Tech-Doc Enabled Items can also reference other Tech-Doc Enabled Items and Business Objects creating a rich Digital Thread. The following sub-sections will provide a brief overview of each of the main features of the Technical Documentation Framework with links to associated sections for more information.

Tech-Doc Enabled Item

This document will refer to an ItemType, or an instance thereof, as being *Tech Doc Enabled* if this ItemType that has been configured to use the Technical Document Editor and store the XML Content in a configured Property. This document will also refer to Items of this type as simply a 'Technical Document'. For the Technical Documentation Application, the `tp_Block` ItemType is an instance of a Tech-Doc Enabled Item.

Structured Content

Technical Documents have content maintained as XML, based on an XML Schema provided by the user. As authors add text and graphics to a document, XML Elements are generated and added by the system and stored in a single Property. The XML provides additional metadata about each

Document Element and the XML Schema ensures that the structure of the document is consistent and compliant.

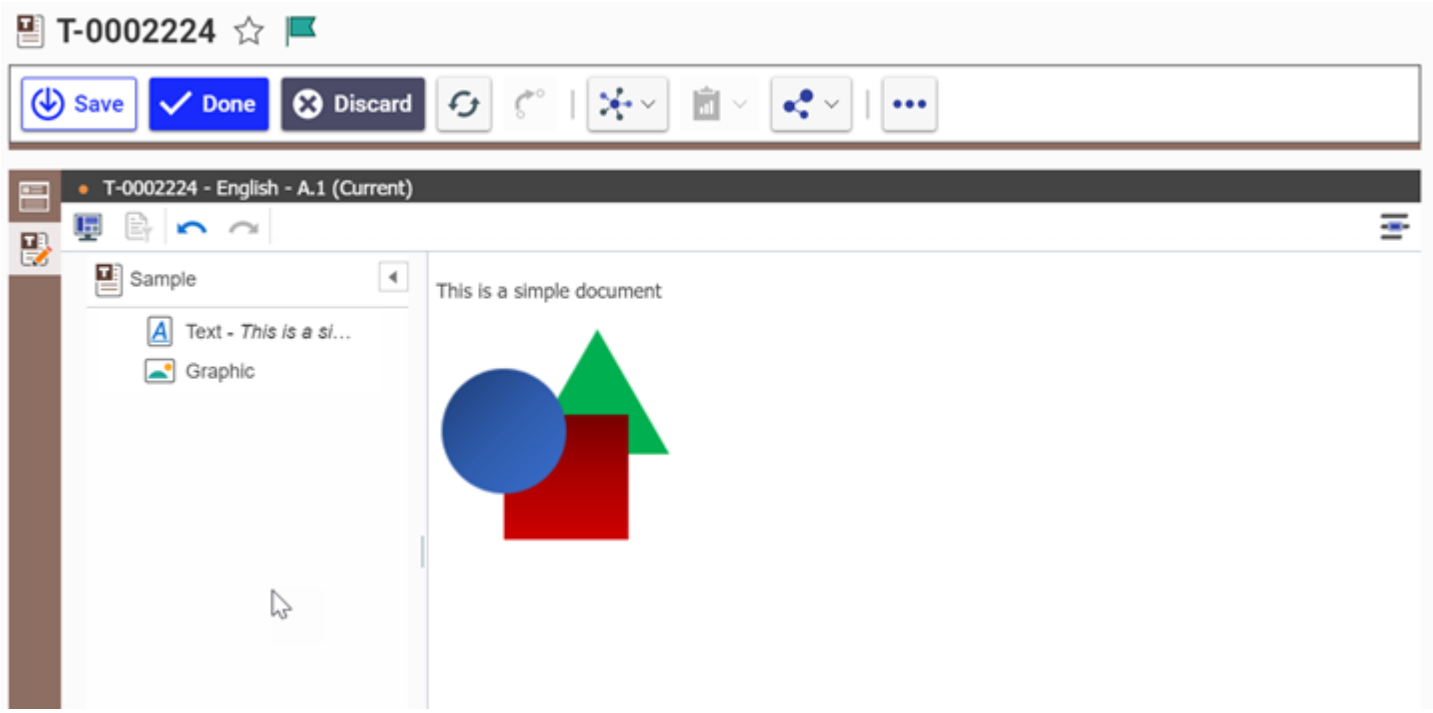


Figure: Sample Document - Showing Editor View

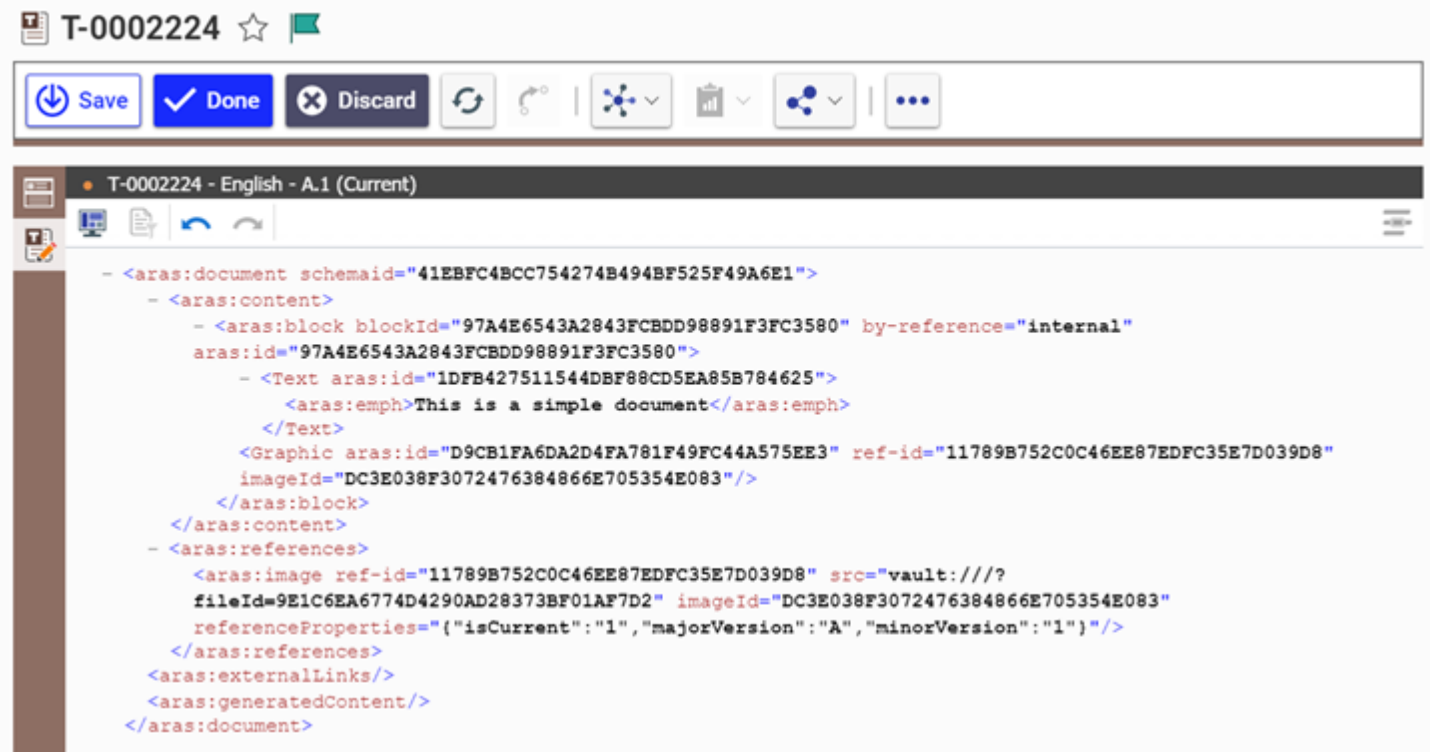


Figure: Sample Document - Showing XML Content View

Document Elements

In XML vernacular, the term ‘Element’ refers to the markup tag (e.g., `<ElementName>`) and everything that is included – as child Elements. This document uses the term ‘**Document Element**’ to identify the types of content that can be added to a Technical Document. Document Elements are stored as individual XML Elements with a name and optional Attributes, which provide additional metadata to be used to further characterize the Document Element, distinguish its style, etc. Document Elements derive from one of the following Types:

- **Container** – Use to *contain* other Document Elements – e.g., Chapter, Section.
- **Formatted Text** – Text that can have basic formatting applied by the author – bold, italic, underline.
- **Unformatted Text** – Text without author-specified formatting
- **List** – Container for List Item Document Types
- **List Item** – Container for content to be displayed in a List.
- **Image** – References a 2D graphic stored by a `tp_Image` ItemType.
- **Table** – Container for Row Document Types

- **Row** – Container for Cell Document Types
- **Cell** – Container for content to be displayed in a cell of a Table.
- **Item** – Container Element that can be associated directly with an Item.
- **Item Property** – Unformatted Text Document Element that references a specific Property on an Item referenced by a parent (or ancestor) Item Document Element.
- **Embedded Document Element** – Refers to any Image, Text, Item, or Item Property Document Element that exists within a parent *Text* Document Element. These *embedded* Document Elements appear in line with surrounding text.

Use of XML Schema and Schema Validation

The authoring process is guided by XML Schema Validation. The Document Elements that can be added at any place in the document is dictated entirely by the XML Schema contained in the associated Document Type and enforced by the Schema Validator. XML Schemas specify the XML Elements, Attributes associated with each, the hierarchy of these elements, and the cardinality. Note that not all the XML Schema *schema* is supported.

Content Generators

In addition to manual addition and removal of Document Elements, a configuration mechanism can be used to execute Content Generators – Method code – to automate the creation/update of Document Elements. This method is typically used when there has been association/reference set between a Document Element and a selected Item Instance whereby Property values from these referenced Items are used specifically to create text or graphics content in the document.

Item Referencing

Technical Documents can have explicit relationships created with other Items maintained in Innovator. These Items are referred to as ‘Business Objects’. This relationship establishes a direct binding between a Document Element and the associated Business Object, although the relationship is maintained by a RelationshipType between the Technical Document and the Business Object. In addition, property data from these Business Objects can be queried and used within the Technical Document. When the Business Object property data is updated, the associated Technical Documents are updated accordingly. This binding helps maintain data integrity and identify a link between Objects and the documentation that was created specifically for them.

Item Reference Configuration

Item Reference Configuration provides a mechanism to automate the creation of values for Properties of Referenced Items and validate content added by the author. Configured JavaScript Methods are used to apply custom business logic and set and/or verify Property Data.

Pick vs Create

When an Item Document Element is created in the Technical Document Editor, the author has the choice of choosing an existing Item to reference or create one at that moment. If an Item is created, the Item Reference Configuration (if defined) is used to prepopulate and/or validate Properties of the generated Item.

Reference ItemTypes

ItemTypes that can be referenced by a Technical Document must be configured as a PolySource for the `tp_Item` ItemType.

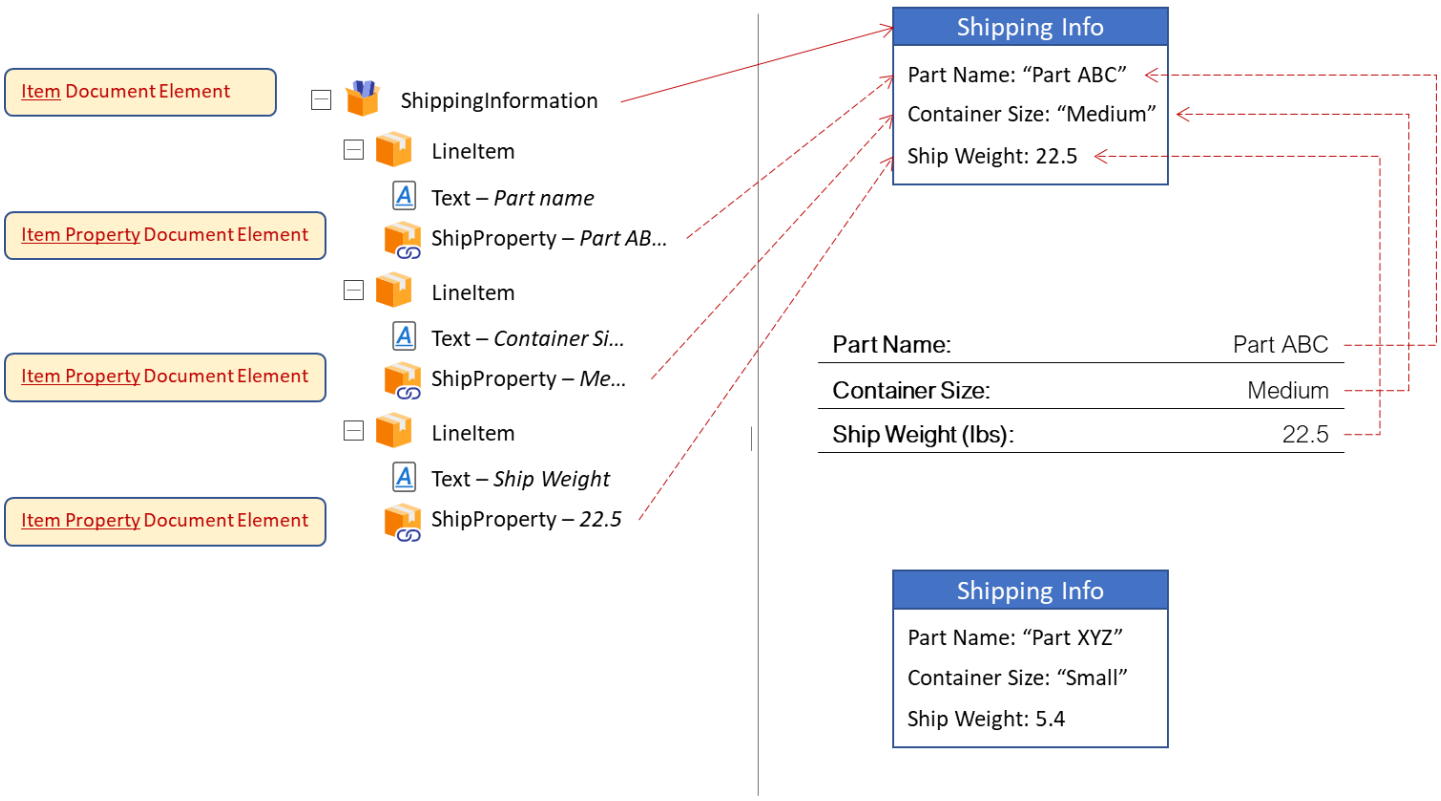
Property References

Document Elements can be configured as a ‘Mapped Property’. These Item Property Document Elements are associated with Properties based on the Property Name. The Item used specifically for Property content is determined by a Document Reference of some Document Element parent (or grandparent, etc.). Using the following illustration as an example, assume that there exists an ItemType – *Shipping Info* – that contains the following Properties:

- Name: *part_name*, Label: *Part Name*
- Name: *container_size*, Label: *Container Size*
- Name: *ship_weight*, Label: *Ship Weight*

In addition, there exists a Technical Document with schema elements:

- ‘ShippingInformation’: Document Element of type ‘Item’ that references a ShippingInformation ItemType and a container for ‘LineItem’s
- ‘LineItem’: Document Element of type ‘Container’ that is a container for one Text Document Element – used for a Label – and one Item Property Document Element – used to reference an Item Property.
- Name: ‘ShipProperty’: Item Property Document Element



In the scenario above, the top-level Item Document Element references a specific Shipping Info Item with the Properties and values shown. Switching this top-level Document Element to reference the other Shipping Info Item would cause the ShipProperty Document Elements to update automatically to the values of this Item.

Filters

Filters provide a mechanism to 'tag' individual Document Elements with metadata ('Filters'), which can then be used to hide content when published. Filters consist of a name and one or more values. For example – a document describing technical information related to an automobile may have a 'Model' Filter that can have values for each of *models* for a particular Make. "A4", "A5", "A6" are example Models for Audi. Filtered Document Elements provide a reuse mechanism to isolate (or hide) certain specific content and expose common content. Using the example, an author would be able to 'publish' a Technical Document filtered by a specific Model.

Document Modules

Technical Documents can exist either as a single entity - with all content contained within – or aggregate content from multiple document *modules*. Breaking up documents into smaller components has multiple advantages including the ability to:

1. Reuse each component.
2. Enable concurrent editing.
3. Apportion each component to areas of specific specialty or expertise.

In the Technical Publication domain, technical content is sometimes referred to as ‘Topics-based’ or ‘Modular’. This refers to content that is about a specific topic or area of focus and is, to some degree, context insensitive. This helps to characterize the content and enable it to exist in multiple places within a document. Contrast this with a narrative in other forms of documents where every paragraph builds upon (and thus depends on) previous paragraphs and context. Each Technical Document instance can evolve independently and be protected by specific access rights. To be reusable, each document component needs to be constructed in a way that allows it to fit within a larger context. To achieve this, the content of each Document / Document Component will be based on an overarching schema or document structure (Section 2.2). This document structure helps ensure consistency and provides a mechanism for determining how individual components can fit into a larger context.

Document Content Styling

The style and layout of Technical Documents is dictated by a separate style configuration that is maintained separately but associated directly with a Technical Document Type definition. Each Document Element defined in the Schema can be referenced and have a unique set of styling parameters that are used to position and render the associated Document Element instances in each document. Style settings can be assigned to HTML output, PDF Output, and the Document Editor. Referencing centrally controlled style settings ensure consistency in look and feel for published content and provides a convenient mechanism to update style in one location and have it automatically applied to all associated documents.

Metadata

All Document Element content can have additional metadata – configured as XML Attributes - assigned to provide additional semantics for content classification and aid in content search and retrieval. This metadata is provided by the Technical Document Author when the Document Structure is designed and configured.

Publication and Data Export

The Technical Documentation application, together with Aras Innovator, provide an authoring and data management environment. To make use of the content created, it needs to be exposed outside the PLM environment. The publishing/export function (Section 0) provides an ability to convert the content of each Technical Document to a format more suitable to end-user consumption.

Templates

Technical Documents can be created to be used as Templates. These ‘Template’ documents can be used to create duplicate content in other Technical Documents – template content is inserted as a copy. The root Document Element of the document identifies where the content of the document can be inserted. Templates are useful when some portion of the content should be updated or extended. Authors should create a reference to other Technical Documents when that content will not be changed.

Images / Graphics

Images used within a Technical Document are managed by the ItemType `tp_Image` . This Item has a File Property for the 2D Image file. Tech-Doc Enabled Items will have a Relationship defined to this ItemType.

Terms

The following Terms are used throughout this document:

Term	Definition
Business Object	Refers to any Item in Innovator that contains information, which may be referred to in a Technical Document.
Cascading Style Sheets (CSS)	Industry standard text-based format for controlling the position, color, size, etc. of web-based content. It is used by the Technical Documentation application to control the format of HTML and PDF content.
Content Filters	Metadata (name/value pairs) that are defined by the Technical Document Administrator and chosen by the Technical Document Author to characterize Document Elements within a Technical Document.
Context Menus	Pop-up menus, used mostly by Document Elements, that contain menu items used to invoke various functions on the Document Element instance. The term ‘context’ is used because the specific menu items displayed in the menu may be different depending on the specific Document Element instance selected.

Document Element	Refers to the components that make up a Technical Document. Every Document Element is associated with a specific node in the Navigation Pane and XML Element in the resulting content. Document Elements are explicitly defined in the Technical Document Type Definition.
Document Style	Refers to the formatting and positional logic that defines the placement and look-and-feel of the rendered content of a Technical Document. Document Style is controlled using Cascading Style Sheets (CSS), associated with a specific Document Type, and defined by the Technical Document Administrator.
Embedded Document Element	Refers to a either a Text, Image, Item, or Item Property that is a direct child of a Text Document Element. When placed within a Text Element, these *embedded* Document Elements are placed in line with surrounding text.
Graphic	Any 2D image that can be displayed in a Technical Document.
Mapped Property	Refers to a Document Element with content that is automatically created from a Property of a referenced Item. Also referred to as an Item Property Document Element.
Structured Content	Refers to the general use of an underlying schema to define the content of a document. Structured Content is contrasted with unstructured (or freeform) content in which case there are no rules that help define and control the content.
Tech Doc Enabled	ItemType that has been configured to use the Technical Document Editor and store the XML Content in a configured Property. The term is also applied to Items (instances) of these ItemTypes.
Technical Document	Refers to instances of a Technical Document ItemType. In this case, the term is capitalized. When lower case, the context is more general – any technical document.