# Xerdi's Documentation Project Product Specification

This document corresponds to package `xdp` version 39ebc70, written on 2024-04-19.

## 1 Product Description

### 1.1 Overview

Xerdi's Documentation Project (XDP) is a powerful system designed to streamline document templating and creation within LaTeX environments. By harnessing the capabilities of LaTeX along with modern document generation techniques, XDP simplifies the process of creating high-quality PDF documents. Its integration-friendly application programming interface (API) and robust document generation capabilities empower users to produce professional documents efficiently and effectively.

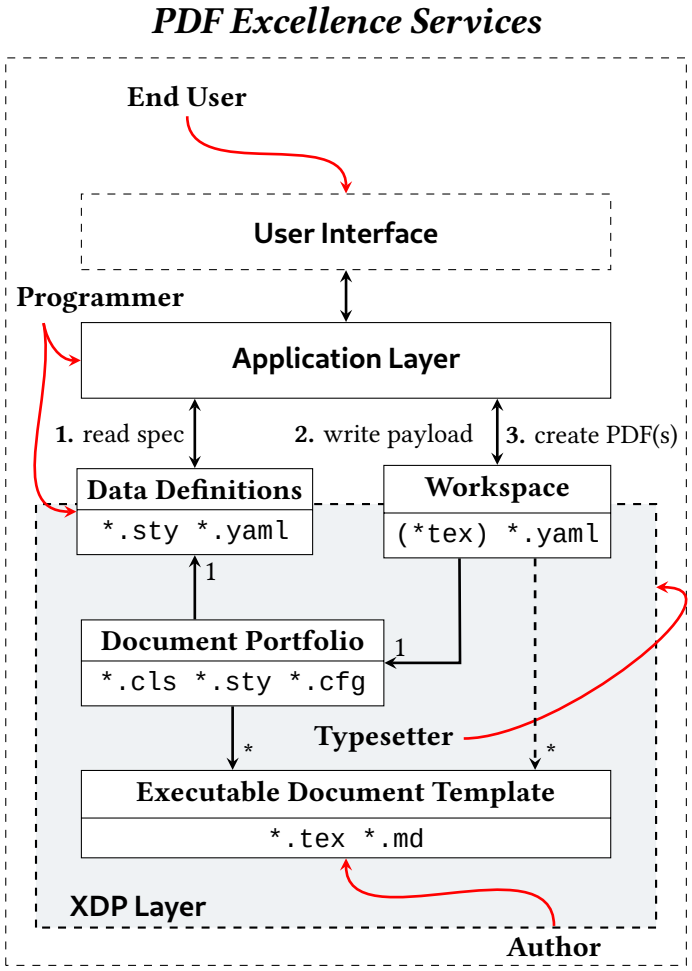


Figure 1: Overview of layers within PDF Excellence and components of XDP

### 1.2 Purpose

XDP serves as a bridge between document template creation and a user-friendly interface for end users, facilitated by a flexible API for programmers. The core purpose of XDP is to enhance document creation workflows by simplifying template creation, facilitating collaboration, and ensuring consistency and quality across documents.

One of the primary objectives of XDP is to empower programmers to create intuitive forms and interfaces that simplify the document creation process for end users[1]. By providing a flexible API, XDP enables programmers to seamlessly integrate customized forms into organizational systems, allowing end users to input data and generate professional documents without requiring expertise in LaTeX or Markdown syntax.

Through collaboration among authors, typesetters, programmers, and end users, XDP ensures that documents meet the highest standards of presentation and functionality. By facilitating the creation of user-friendly forms and interfaces, XDP enhances productivity, efficiency, and user satisfaction in the document creation process.

[1] Refer to figure 1 for a clear distinction of XDP's scope.

# 2 User Profiles

## 2.1 Authors

**Description** Authors are responsible for creating the initial version of document templates using XDP. They play a crucial role in content creation, ensuring clarity, coherence, and adherence to editorial guidelines.

**Key Responsibilities**

- Draft document templates efficiently using LaTeX or Markdown.
- Ensure the quality and consistency of written content, including grammar, style, and formatting.
- Collaborate with typesetters and programmers to refine document templates and interfaces.

**Skills and Requirements** Basic understanding of document structuring and formatting, familiarity with LaTeX or Markdown syntax, and proficiency in content editing and writing.

**Training and Support**

- Basic Authoring Training (Markdown)
- Professional Authoring Training (LaTeX)

## 2.2 Typesetters

**Description** Typesetters specialize in formatting and styling documents to meet specific standards and specifications. They collaborate with authors to ensure consistency in document layout, typography, and design.

**Key Responsibilities**

- Refine document layout and formatting to meet specified standards.
- Design document interfaces for executing document templates seamlessly.
- Ensure adherence to aesthetic preferences and document standards.

**Skills and Requirements** Proficiency in typesetting and formatting using LaTeX, familiarity with document design principles, and strong attention to detail.

**Training and Support**

- Document Portfolio Management Training (LaTeX, Git)
- Document Interfacing Training (LaTeX, YAML)
- Document Class Creation Training (LaTeX)
- Markdown Compliant EDTs Training (LaTeX, Markdown)

## 2.3 Programmers

**Description** Programmers integrate document creation into organizational workflows and systems using XDP's flexible API and automation capabilities.

**Key Responsibilities**

- Streamline document generation processes and integrate with existing software solutions.
- Design document interfaces for executing document templates and facilitate seamless workflow integration.
- Maintain and enhance the Application Layer for efficient communication between users and the system.

**Skills and Requirements** Proficiency in software development, familiarity with API integration, and understanding of document management systems.

**Training and Support**

- Document Compilation & Workspace Management (LaTeX compilers, Git)
- Document Interfacing Training (YAML)

## 2.4 Collaboration and Interaction

Authors, typesetters, and programmers collaborate closely throughout the document creation process. Authors provide content and draft document templates, typesetters refine document layouts and styles, and programmers integrate document creation into organizational systems.

## 2.5 Tools and Resources

Users have access to training programs, documentation, and support resources provided by Xerdi's PDF Excellence Services. Additionally, XDP offers a user-friendly interface and integration-friendly environment to streamline collaboration and document creation.

## 2.6 User Feedback and Improvement

Xerdi encourages user feedback to continuously improve XDP. Users can provide feedback through various channels, including forums, documentation, or customer support channels.

## 2.7 User Satisfaction and Success Metrics

User satisfaction and success metrics are assessed based on productivity, efficiency, user experience, and the quality of output. Xerdi monitors these metrics to ensure that XDP meets the needs and expectations of its users.

# 3 Key Features

## 3.1 Suitability for Paper Office, Paperless Office, or Hybrid

XDP is suitable for organizations transitioning between paper-based and paperless document management systems, as well as those embracing hybrid approaches. It offers flexibility to accommodate varying organizational needs and workflows, facilitating smooth transitions and ensuring compatibility with existing practices.

## 3.2 Compatibility with AI Co-Writing

XDP supports input formats (LaTeX and Markdown) that are suitable for co-writing with AI, such as ChatGPT 3.5 or higher. This capability enhances collaboration and productivity by allowing users to leverage AI assistance in drafting documents.

## 3.3 Streamlined Document Template Creation

XDP simplifies document template creation by allowing the typesetter to provide intuitive tools for authors to design custom layouts, styles, and content structures. Authors can easily create templates tailored to their specific needs, whether using LaTeX or Markdown. This approach ensures that authors are offered a distraction-free way of writing, while benefiting from the typesetter's expertise in creating generic document classes.

## 3.4 Dynamic Placeholder Management

With XDP, authors can insert dynamic placeholders to abstract user input complexities, streamlining the document creation process. These placeholders allow for flexible content insertion and customization, ensuring adaptability and ease of use.

## 3.5 Integration-Friendly Environment

XDP offers a user-friendly and integration-friendly environment, enabling seamless collaboration among authors, typesetters, and programmers. Its flexible API facilitates smooth integration into organizational systems, ensuring compatibility and interoperability.

## 3.6 Efficient Document Generation

Engineered for efficiency, XDP ensures fast and reliable document generation. Authors can effortlessly generate PDF documents from custom templates, saving time and effort while maintaining high-quality output.

## 3.7 Robust Version Control

With built-in Git integration, XDP enables robust version control for document management. Organizations can maintain a transparent change history, track document revisions, and collaborate effectively, ensuring accountability and traceability throughout the document creation process.

# 4 User Requirements

## 4.1 Functional Requirements

FR.1 Authors should be able to create content in either LaTeX or Markdown format.

FR.2 Authors should have the capability to define and manage placeholders within templates to abstract user input. Placeholder management, including administration and integration, should be facilitated through YAML, allowing for seamless integration with document content.

FR.3 Programmers should be able to seamlessly integrate document templates with their YAML specifications into the organization's document creation workflow.

FR.4 Users should have the ability to effortlessly generate PDF documents from the customized templates.

FR.5 The system must seamlessly integrate with Git for version control, allowing users to efficiently track document revisions and collaborate on document creation.

FR.6 Authors must be able to operate offline, ensuring continuous productivity even in environments without internet connectivity. Offline operation should provide access to essential features for document creation and management.

## 4.2 Non-Functional Requirements

NFR.1 Performance - XDP should efficiently generate PDF documents within a reasonable time frame, regardless of the complexity or size of the templates.

NFR.2 Reliability - The system must maintain consistent performance levels and uptime, with minimal disruptions or failures during document generation processes.

NFR.3 Scalability - XDP should seamlessly scale to accommodate an increasing number of users and documents, ensuring optimal performance under varying workloads.

NFR.4 PDF/A-1b Compliance - Every document generated by XDP must adhere to the PDF/A-1b standard, guaranteeing long-term preservation and accessibility of the documents in compliance with archival requirements.

NFR.5 Auditability - All changes made within XDP components must be traceable using Git, enabling comprehensive auditing and ensuring accountability for modifications.

NFR.6 Document Signing - Every PDF document produced by the system needs to be signed detached using GPG, ensuring document authenticity and integrity.

# 5 Technical Requirements

TR.1 Operating System Compatibility: XDP should be compatible with major operating systems, including Windows, macOS, and Linux, ensuring seamless accessibility for authors across diverse platforms.

TR.2 Scalable Code Architecture: The system must be developed using a scalable and maintainable code architecture, facilitating flexibility for future updates and enhancements while ensuring long-term stability and efficient resource utilization.

TR.3 Seamless System Integration: XDP should provide an interface defined and served with YAML for seamless system integration, promoting interoperability with other software tools and platforms and enhancing overall workflow efficiency.

TR.4 Workstation Environment Requirement: The workstation environment for authors and typesetters must meet the following criteria:

a) It must include an integrated development environment (IDE) with LaTeX support for creating, customizing, and maintaining document templates. Recommended IDE options include PyCharm with the TeXiFy Idea plugin and Visual Studio Code (VS Code) with the LaTeX Workshop Extension.

b) Additionally, it must include a PDF viewer capable of detecting file changes in the PDF documents generated by XDP. This functionality ensures efficient document drafting by providing real-time feedback on document changes.

TR.5 Git Integration for Version Control: The system must seamlessly integrate with Git for version control, utilizing Git hooks and APIs for efficient document collaboration and revision tracking. This integration enhances team productivity and streamlines document management processes, ensuring accurate version control and effective collaboration among users.

TR.6 Robust Error Handling and Logging: XDP should implement robust error handling and logging mechanisms to facilitate troubleshooting and debugging, ensuring smooth operation and minimal disruptions for users. Comprehensive error reporting and logging enhance system reliability and maintainability, enabling timely resolution of issues.

# 6 Data Requirements

DR.1 Document Content Versatility - XDP should support various data formats for document content, including plain text, images, tables, and mathematical equations, ensuring versatility in document creation.

DR.2 Dynamic Hyperlinks - Hyperlinks within documents should be placed by label, not hard-coded, ensuring flexibility and ease of maintenance.

DR.3 Secure Storage Options - The system must securely store user-generated templates and documents, offering options for both local storage and cloud-based solutions to accommodate user preferences and organizational requirements.

DR.4 Robust Backup and Recovery - The system must implement robust backup and recovery mechanisms to mitigate the risk of data loss due to system failures or unforeseen events, safeguarding user data and document content.

DR.5 Data Privacy Compliance - XDP should adhere to data privacy regulations and standards, prioritizing the protection of sensitive user data and document content, thereby fostering trust and confidence among users regarding data security and confidentiality.

# 7 Constraints

C.1 Licensing Compliance - XDP must comply with licensing restrictions for any third-party libraries or components used in its development to ensure legal and ethical use of software resources.

C.2 Resource Limitations - The system must operate within the resource limitations of the target deployment environment, including memory, disk space, and processing power, to maintain optimal performance and stability.

C.3 Budget and Timeline Adherence - XDP development must adhere to a predefined budget and timeline, with constraints on allocated resources and manpower, to ensure project feasibility and timely delivery.

C.4 Compatibility Requirement - The system must be compatible with existing organizational infrastructure and technologies, including authentication mechanisms and data storage solutions, to facilitate seamless integration and interoperability.

C.5 Backward Compatibility - XDP must support backward compatibility with previous versions of document templates and formats to ensure seamless migration for existing users and preserve legacy data integrity.

C.6 Cross-Platform Compatibility - Every component or package within XDP must maintain its cross-platform compatibility and be compatible with the latest version of TEX Live, ensuring consistent performance and accessibility across different operating systems.

# 8 Assumptions

A.1 Authors' Proficiency - Authors of XDP are assumed to have a basic understanding of LaTeX or Markdown syntax for document creation and customization, enabling effective utilization of the system's features.

A.2 Programmers' Proficiency - Programmers of XDP are assumed to have a basic understanding of HTML form creation, working with YAML files, and command-line features, facilitating document template execution for the end user.

A.3 Typesetters' Proficiency - Typesetters utilizing XDP are assumed to have expertise in formatting and styling documents using LaTeX and related tools, ensuring high-quality document presentation and layout.

A.4 Internet Connectivity - The target deployment environment for XDP is assumed to have reliable internet connectivity for accessing cloud-based services or online resources, such as version control repositories, ensuring seamless access to external dependencies.

A.5 Device Compatibility - XDP users are assumed to have access to modern web browsers or compatible devices for accessing the system's user interface and functionality, ensuring optimal user experience across different platforms.

A.6 Maintenance and Updates - The system is assumed to receive regular maintenance and updates to address security vulnerabilities, performance issues, and user feedback, ensuring continuous improvement and reliability.

A.7 User Authorization - XDP users are assumed to be authorized and authenticated individuals with appropriate permissions to create, edit, and manage document templates and content within the system, maintaining data security and access control.

# 9 Acceptance Criteria

AC.1 Template Creation: The system must allow users to create custom document templates using LaTeX or Markdown syntax, providing comprehensive support for defining layout, styles, and content structures.

AC.2 Placeholder Management: XDP should offer a user-friendly interface for managing placeholders and dynamic content insertion, enabling users to abstract user-input complexities during document creation effectively.

AC.3 Collaboration and Version Control: The system should support seamless collaboration among users through decentralized, local drafting capabilities and robust Git integration, ensuring efficient version control and document synchronization.

AC.4 Deployment Ease: XDP's Docker image should be easily deployable across various operating systems and environments, accompanied by straightforward setup instructions and compatibility with common infrastructure configurations.

AC.5 Comprehensive Documentation: The system's documentation must be comprehensive, covering all features, functionalities, and usage guidelines, with clear examples and tutorials catering to users of varying expertise levels.

AC.6 Change Management: XDP's integration with Git must enable organizations to maintain a transparent change history and generate detailed change logs, ensuring accountability and traceability throughout the document creation process.

AC.7 Performance and Scalability: The system's performance should meet predefined benchmarks and performance metrics, including response times, document generation speed, and scalability under varying workload conditions.

AC.8 Security Compliance: XDP must comply with security best practices and standards, encompassing data encryption, access controls, and vulnerability management, to ensure the confidentiality, integrity, and availability of user data.

AC.9 Workflow Integration: The system should seamlessly integrate with existing organizational workflows and systems, facilitating easy adoption and interoperability with other software solutions.

AC.10 User Interface Accessibility: XDP's user interface must be intuitive and accessible, offering support for multiple languages and customization options to accommodate diverse user preferences and requirements effectively.

# Glossary

**Git** Git is a distributed version control system used for tracking changes in source code during software development. It allows multiple developers to collaborate on projects simultaneously by managing revisions, facilitating branching and merging, and maintaining a complete history of changes. Git enables developers to work efficiently in both local and remote environments, providing mechanisms for code review, issue tracking, and continuous integration. With its flexibility, speed, and robustness, Git has become a cornerstone tool in modern software development workflows. 2, 4–6

**Git hooks** Git hooks are customizable scripts that Git executes before or after certain key actions, such as commits, merges, or pushes, to automate or enforce specific workflows and behaviors. These scripts allow developers to integrate custom logic or enforce project-specific policies during the Git workflow. Git hooks are stored in the `.git/hooks` directory of a Git repository and are triggered automatically by Git events. They can be written in any scripting language, such as Bash, Python, or Perl, and can perform a wide range of actions, including code linting, running tests, enforcing coding standards, sending notifications, or triggering deployment processes. By leveraging Git hooks, development teams can enforce consistency, improve collaboration, and automate repetitive tasks within their version control workflows. 6

**LaTeX** LaTeX is a high-quality typesetting system commonly used for producing technical and scientific documents such as academic papers, reports, and books. It provides precise control over document layout and formatting, allowing users to focus on content creation rather than formatting. LaTeX uses markup commands to define document structure and formatting elements, making it highly customizable and flexible. With its robust features and extensive package ecosystem, LaTeX is a preferred choice for professionals in various fields seeking professional-looking document outputs. 2–4, 8

**Markdown** Markdown is a lightweight markup language that allows users to format plain text documents using simple, intuitive syntax. It is widely used for writing content that will be converted into HTML for web publishing. Markdown offers a straightforward way to add formatting elements such as headers, lists, links, and emphasis to text without the complexity of traditional markup languages like HTML. Markdown documents are easy to read in their raw form and can be converted into various formats such as HTML, PDF, and Word documents using specialized tools and converters. Its simplicity and versatility make Markdown a popular choice for creating documentation, writing blog posts, and collaborating on text-based projects. 2, 4, 5, 8

**TeX Live** TeX Live is a comprehensive distribution of the TeX typesetting system, including LaTeX, ConTeXt, and related programs and macro packages. It provides a complete environment for producing high-quality typeset documents, ranging from academic papers and technical reports to books and presentations. TeX Live ensures consistency and compatibility across different platforms, offering a vast collection of packages and fonts

for various typesetting needs. With regular updates and maintenance, TeX Live stays up-to-date with the latest developments in the TeX community, making it the preferred choice for many users in academic, scientific, and publishing domains. 7

**XDP** Xerdi's Documentation Project (XDP) is a comprehensive system designed to simplify document creation and management within LaTeX environments. It offers a user-friendly interface and powerful document generation capabilities, catering to the needs of authors, typesetters, and programmers alike. XDP empowers users to create high-quality PDF documents efficiently with flexible document template creation tools. It streamlines document drafting and collaboration with decentralized, local drafting capabilities and seamless Git integration. Additionally, XDP provides extensive documentation and a wide array of document classes, packages, and tools to meet diverse document requirements. 1–9

**YAML** YAML, short for "YAML Ain't Markup Language," is a human-readable data serialization format commonly used for configuration files and data interchange. YAML files are designed to be easily readable by both humans and machines, using a simple and intuitive syntax based on indentation and key-value pairs. YAML supports a wide range of data structures, including lists, dictionaries, and scalar values, making it versatile and expressive for representing complex data hierarchies. It is often used in software development, system administration, and other fields where structured data needs to be stored, exchanged, or manipulated. YAML's simplicity, readability, and wide adoption make it a popular choice for configuration management and data serialization tasks. 2, 3, 5, 6