

Dynamic PI-Classification and UML Generation for Automotive Manual Intelligence [†]

Minnah Aamir ¹ and Debopriyo Roy ^{2,*} 

¹ Information Technology and Project Management, Technical Communication Laboratory, Center of Language Research, The University of Aizu, Aizuwakamatsu 965-0028, Fukushima, Japan; m5291501@u-aizu.ac.jp

² Technical Communication Laboratory, Center of Language Research, The University of Aizu, Aizuwakamatsu 965-0028, Fukushima, Japan

* Correspondence: droy@u-aizu.ac.jp

[†] Presented at the 8th International Global Conference Series on ICT Integration in Technical Education & Smart Society, Aizuwakamatsu City, Japan, 20–26 January 2026.

Abstract

As industries transition to Industry 4.0 and 5.0, the reliance on intelligent information systems continues to grow. However, many technical manuals remain static, text-based documents that are difficult to search, interpret, and apply in real-world operational contexts. This paper presents Smart Manual Assistant, an automated conversational system that transforms complex automotive manuals into interactive, role-adaptable knowledge resources. The core of the system leverages a Large Language Model (LLM) to dynamically generate PI-Class metadata, organizing content according to key intrinsic and extrinsic product and information attributes. This structured metadata enables precise content classification and supports context-aware responses tailored to different user types, including drivers and service technicians. A key contribution of the system is its dual-output response mechanism. Alongside natural language explanations, the assistant automatically generates PlantUML code and the corresponding visual diagrams (e.g., activity, sequence, and component diagrams). This provides immediate visualizations of system interactions and procedural workflows, significantly improving comprehension and usability. The Smart Manual Assistant implements a complete end-to-end pipeline—from manual ingestion to intelligent, visual, and role-specific guidance. The solution is built using a Django REST backend, a Chatbase-based interface, and Zapier automation for conversation logging to Google Sheets. The results demonstrate a practical framework for transforming static technical documentation into adaptive, data-driven support systems, enhancing accessibility, training efficiency, and real-time decision-making.

Keywords: PI-classification; automotive documentation; UML generation; Large Language Models; semantic information access; Industry 4.0



Academic Editors: George Fragulis and Peter Ilic

Published: 20 July 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

The accessibility of technical documentation has always been a major challenge for the automotive industry. Hundreds of diverse systems, components, structures and versions with interconnected functions make up modern vehicles and are documented by very long user manuals that often exceed hundreds of pages. These lengthy documents form very critical barriers to effective information retrieval by common drivers in need of quick answers to operational questions and service technicians diagnosing complex faults. In the Industry 4.0–5.0 landscape of intelligent manufacturing and intelligent systems, technical

documentation also has to go beyond static PDF repositories into systems that can provide instant, contextual, and role-appropriate information delivery, turning static documents into active knowledge assistants using PI classification [1]. PI Classification, introduced by Wolfgang Ziegler, is a semantic structuring methodology used in technical communication to classify manual content according to two key dimensions of information: (1) P (Product Classes)—describing what the content is about—and (2) I (Information Classes)—describing how the information is used. This framework separates product context (what part of the system is being discussed) from information purpose (why the user needs it).

This is particularly critical in automotive contexts, where several challenges arise. Time-sensitive decisions often require immediate access to procedural information, particularly in situations involving emergency troubleshooting [2]. Additionally, the cognitive load associated with complex repairs underscores the need for simplified and visually guided instructional support [3]. Furthermore, variability in user skill levels—ranging from novice drivers to experienced technicians—necessitates adaptive content presentation to ensure clarity and accessibility [2]. Finally, information fragmentation across multiple manual sections can impede holistic understanding of system interactions, reducing overall maintenance efficiency [3].

Traditional approaches to manual digitization—such as keyword search, table-of-contents navigation, or basic chatbot interfaces—fail to capture the semantic relationships between information types, user intent, and procedural context. Ontology-based frameworks (e.g., OWL) can formally represent such relationships, but they require extensive domain modeling effort and are often difficult to maintain in dynamic industrial settings [4]. To address this gap, the Smart Manual Assistant employs a Large Language Model (LLM) capable of implicitly inferring semantic associations through its learned latent representations, allowing context-sensitive retrieval without relying on a fully engineered ontology structure [5]. This integration of LLM-driven reasoning with structured classification methodologies supports more adaptive, role-aware, and procedurally relevant information delivery.

This paper addresses this gap by introducing the structure for a Smart Manual Assistant that automates the transformation of automotive manuals into an interactive, intelligent knowledge base. The system's contributions are threefold:

1. **Dynamic PI Classification:** Using an LLM-driven pipeline to automatically tag every user query with four-dimensional PI-Class metadata, which enables semantic information retrieval. In this system, PI Classification enables the LLM to:
 - (a) Identify the type of question (informational vs. procedural vs. diagnostic).
 - (b) Identify the relevant product component or subsystem.
 - (c) Generate responses that are precisely scoped and role-adapted.
2. **Role-Aware UML Generation:** When a Large Language Model is fed PI-classification metadata, it can dynamically select the appropriate UML diagram type (Activity, Sequence, Use-Case, etc.) and generate both the diagram source (e.g., PlantUML) and a human-readable explanation that is tailored to the user's role and the semantic class of the query. This integrated systemic approach—Role-Aware UML Generation—produces visualizations that match the user's cognitive needs and expertise level: simplified, high-level activity or use-case diagrams for drivers; and detailed sequence or component diagrams for technicians.
3. **End-to-End Automation for user:** Integrated architecture that connects a conversational frontend provided by Chatbase with a Django REST backend, Gemini LLM, and Zapier for analytics logging, all connected via a seamless workflow.

The basic design of the Smart Manual Assistant that we worked with is shown in Figure 1.

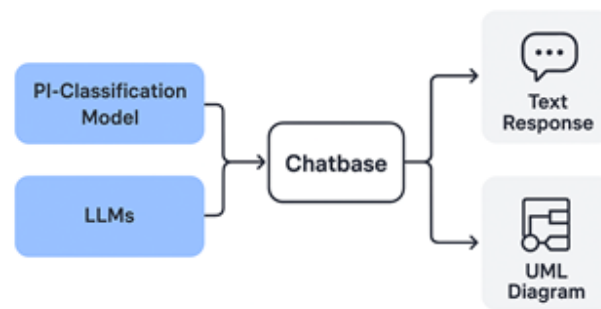


Figure 1. Smart Manual Assistant Design.

2. Background and Theoretical Framework

This section establishes the theoretical and practical foundations of our work. This paper first reviewed the PI-Classification framework as a method for structuring technical information, and then examined the role of visualizations, specifically UML, in enhancing comprehension of complex systems.

2.1. The PI-Classification Schema: A Metadata Framework for Technical Information

The challenge of managing technical documentation for complex, variant-rich products like automobiles necessitates a structured, metadata-driven approach [1]. The PI-Class method provides a rigorous framework for classifying modular content to facilitate its creation, management, and reuse within Content Management Systems (CMS). This methodology is built upon a dual-axis metadata schema, separating product-specific (P) from information-specific (I) criteria, with each axis further divided into intrinsic and extrinsic dimensions.

Intrinsic metadata defines the core, invariant identity of a content module. Extrinsic metadata describes the variable contexts in which that module can be applied. This creates a powerful four-dimensional classification space, as defined in Table 1.

Table 1. The Four Dimensions of the PI-Classification Schema.

Dimension	Category	Description	Automotive Example
Product (P)	Intrinsic	The physical or virtual component the information is directly about.	Engine, TPMS Sensor, Braking System
	Extrinsic	The product hierarchy, model family, or variant.	Toyota Camry 2022, All Hybrid Variants
Information (I)	Intrinsic	The inherent type of information or its communicative purpose.	Procedural, Diagnostic, Safety, Descriptive
	Extrinsic	The document type, output medium, or target audience.	Quick-Start PDF, Workshop Manual, Infotainment Screen Tip

This structured schema allows content modules to be uniquely defined by their intrinsic values (e.g., a diagnostic procedure for the engine) while being flexibly reused across extrinsic contexts (e.g., in both the Toyota Camry and Lexus ES workshop manuals).

2.2. Traditional vs. Dynamic Application of PI-Class

In conventional content management workflows, the application of PI-Class metadata is a manual, upfront process conducted by technical writers during the authoring phase within a CMS. Content modules are tagged and stored, and this static metadata is used for

retrieval and variant management when publishing predefined documents. While effective for content reuse, this paradigm is inherently limited: the classification is locked to the content object and cannot adapt to the specific context or intent of an individual user's real-time query. Our work introduces a paradigm shift by repurposing PI Classification as a dynamic, real-time interpretation layer. We leverage a Large Language Model (Gemini 2.0 Flash) to perform on-the-fly PI-Class analysis of natural language queries. When a user submits a query, the LLM does not simply search for keywords. Instead, it acts as a semantic parser, generating the corresponding PI-Class metadata for the query itself.

This transformation of PI-Classification from a static content attribute into a dynamic intent-understanding mechanism is a core innovation of our system. It enables precise, context-aware semantic information retrieval, moving beyond the limitations of traditional CMS to meet the on-demand needs of users in the field.

2.3. UML as a Visual Language for Technical Communication

Understanding complex procedures and system interactions remains a central challenge in automotive technical communication. Visual representations have been shown as in Figure 2 to reduce cognitive load, enhance information retention, and support faster task execution [3]. In this context, the Unified Modeling Language (UML) provides a standardized and expressive visual notation well-suited for depicting the mechanical, electrical, and software interactions present in modern vehicle systems [6].

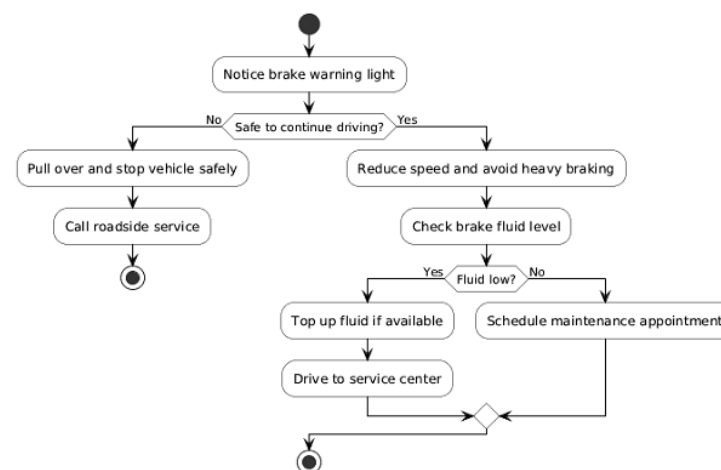


Figure 2. UML Diagram Type (Activity Diagram).

In the proposed Smart Manual Assistant, PI-classification provides structured meta-data that identifies the product elements involved and the purpose of the information (e.g., diagnostic, procedural, or descriptive) [3]. When combined with a Large Language Model (LLM) capable of interpreting user intent and role (e.g., service technician vs. driver), the system can automatically select the most appropriate UML diagram type—such as Activity, Sequence, or Use Case—and generate it dynamically.

This role-aware UML generation allows technicians to view detailed component interactions and diagnostic workflows, while drivers receive simplified procedural or safety-oriented visual guides as mentioned in the Table 2 below. Thus, the integration of PI-induced content structuring with LLM-driven semantic inference provides a scalable method for producing clear, context-sensitive visualizations that improve comprehension and support more efficient task performance [1,5].

Table 2. UML Diagram Types and Their Application Contexts.

UML Diagram Type	Purpose	User Role	Application Context
Use Case Diagram	Identify system actors and their goals.	Drivers and Technicians	Clarifies who interacts with the vehicle system and why
Activity Diagram	Show procedural steps and decision paths	Drivers	Provides simplified workflows for safety and confidentiality.
Sequence Diagram	Represents communication between system components	Technicians	Supports diagnostic reasoning and signal flow analysis.
Component Diagram	Depict physical or logical system structure	Technicians	Helps identify part relationships and failure inquiries.

While the potential of UML for technical documentation has been recognized, its incorporation has largely remained a manual process, requiring significant expertise and effort from both domain and modeling experts. This has limited its widespread adoption in dynamic support systems. Our system addresses this bottleneck by automating the translation. By leveraging the structured understanding derived from the dynamic PI-Classification, our LLM generates accurate PlantUML code, transforming dense textual descriptions into intuitive visual guides instantly. This automation makes complex automotive intelligence immediately more accessible and actionable for a diverse range of users, directly fulfilling the promise of intelligent, on-demand information delivery.

3. Dynamic PI-Classification Semantic Information Access

Unlike traditional PI-Class systems that rely on manually applied, static metadata, our approach dynamically generates PI-Class classifications in real time using a Large Language Model (LLM), transforming each user query into a structured semantic representation of intent and context, enabling intelligent and role-aware responses—and the following section explains how this dynamic classification process is implemented and integrated into the Smart Manual Assistant workflow.

3.1. The Four-Dimensional Metadata Model

The semantic understanding of our system is anchored in the real-time generation of PI-Class metadata. For every user query, the Gemini 2.0 Flash LLM is prompted to analyze the text and return values for the four dimensions. The prompt instructs the LLM to act as a technical information classifier, using definitions and examples of Intrinsic or Extrinsic Product and Information classes.

3.2. Use Case 3: User—Bluetooth Connection Feature

Let us see an example of a use case with a UML diagram in the Table 3 explaining how the activities unfold in real-time.

Table 3. A Use Case With Bluetooth.

Figure Element	PI-Class Category	Type	Information Class
User action “Turn on Bluetooth”	Procedural	Intrinsic	Intrinsic Information
“Go to Settings” or “Bluetooth Enabled Successfully”	Product Behavior (Bluetooth Function)	Intrinsic	Intrinsic Product
“Confirmation” Display		Extrinsic	Extrinsic Information
Bluetooth Connection Content	Product Subsystem	Extrinsic	Extrinsic Product

The selected UML Diagram could be an activity Diagram. The dynamic tagging transforms a vague request into a precise, structured query that the system can reason with.

3.3. Semantic Simplification Layer

The detailed four-dimensional PI-Class is too complex for direct use. We simplify it into nine clear categories as mentioned in Table 4 to make the system easier to manage, analyze, and understand. This creates a bridge between the complex classification and the final user application.

Table 4. PI-Class to Simplified Category Mapping.

Simplified Category	PI-Class Context Description	Simplified Category	PI-Class Context Description
Maintenance	Cleaning, lubrication, inspection, or general upkeep	Safety	Warnings, indicators, immobilizers, airbags, seat belts
Diagnostics	Fault finding, troubleshooting	Indicators	Dashboard or warning lights
Controls	Switches, levers, infotainment interfaces	Electrical	Wiring, sensors, relays, ECU details
Engine	Combustion, cooling, fuel systems	Transmission	Clutch, gear, drivetrain
Other	Miscellaneous items not covered by other categories		

4. The Ontology Matrix for Semantic Access

To operationalize the classification, we developed an ontology matrix explained in Table 5 that connects the simplified category, user role, and question type to the intended system behavior. This matrix ensures the response is not just accurate, but also contextually and pedagogically appropriate, fulfilling the promise of semantic access.

Table 5. Ontology Matrix in Wide Format.

Category	User Role	Question Type	Intended System Behavior & Response Focus
Diagnostics	Technician Driver	How or Why What or Why	Diagnostic logic, component interaction, root-cause analysis Warning explanation, safe actions, call for service
Maintenance	Technician Driver	How How	Tool-specific procedures, torque specs, safety standards Step-by-step instructions, common tools, clear warnings
Infections	Driver	What	Symbol's meaning, cause, and driver response
Controls	Driver	How	Direct procedure for feature

This ontology demonstrates how the classification directly “feeds into semantic access” by allowing the system to tailor the depth, focus, and presentation of information based on a deep understanding of the user’s intent and context.

5. System Architecture and Implementation

The structured understanding derived from the PI Classification is directly leveraged to generate the most appropriate visual aid. This section outlines the rules for diagram selection and how complexity is adapted for different users.

The Diagram Selection Ontology

The selection of a UML diagram type is a deterministic decision based on the output of the dynamic PI-classification explained in Table 6. The primary drivers are the User Role and the Intrinsic Information Class.

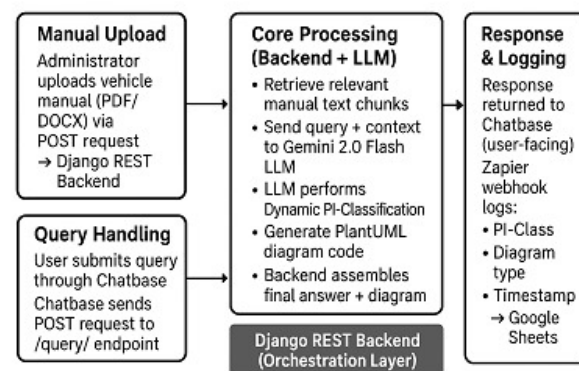
Table 6. UML Selection Logic.

User Role	Information Class	UML Diagrams	Rationale
Driver	Procedural	Activity Diagram (Basic)	Linear sequence of driver actions
Technician	Procedural	Activity Diagram (Detailed)	Branches, tool steps, quality checks
Technician	Diagnostic	Sequence Diagram	Component interaction to isolate faults
Driver or Technician	Descriptive	Component Diagram	System structure and relationships
Driver	Conceptual	Use Case Diagram	High-level user interaction

6. Evaluation and Use Case Analysis

The theoretical model is realized through a fully automated, integrated pipeline. Figure 1 illustrates the end-to-end workflow, with the LLM serving at its center.

Figure 3 explained the process clearly.



Step	Component	What Happens
1	User (Driver/Technician)	Submits a question through the Chatbase interface
2	Chatbase -> Django Backend	The question is forwarded to the backend for processing
3	Document Retrieval	Backend fetches relevant text chunks from the uploaded manual
4	LLM (Gemini 2.0 Flash)	Performs Dynamic PI-Classification to understand product information type, and user role
5	LLM Output Generation	Produce both: (1) Text answer and (2) PlantUML code
6	Backend	Renders PlantUML into final UML diagram image
7	Response Sent Back to User	Chatbase displays the final answer + UML diagram
8	Zapier Log to Google Sheets	Systems stores metadata for analytics and evaluation

Figure 3. The Workflow Model—Smart Manual Assistant Process Flow.

- Manual Upload:** An administrator uploads a car manual (PDF or DOCX) via a POST request to the Django REST backend endpoint using Postman. The backend parses and stores the text in the database.
- Query Handling:** A user submits a query via the chat-based interface. Chatbase, via a pre-configured custom action, sends a POST request to the backend's query endpoint.
- Core Processing:** The Django backend orchestrates the core intelligence:
 - It retrieves relevant text chunks from the manual.
 - It sends a query and context to Gemini 2.0 Flash LLM.
 - The LLM executes the Dynamic PI-Classification and generates the corresponding PlantUML code based on the schema in Table 2.
 - The backend compiles the final response (text answer + image URL of the rendered diagram).

4. **Response and Logging:** The response is sent back to Chatbase for the user. Simultaneously, the backend triggers a Zapier webhook, logging the entire transaction, including the PI-Class metadata, diagram type, and timestamp, into a Google Sheet for analytics.

This seamless workflow demonstrates a complete, functioning system for intelligent information delivery.

7. Use Case Analysis and Discussion

To validate the system's efficacy and its role-aware functionality, we present two detailed use cases from the system's transaction log.

7.1. Use Case 1: Technician—Diagnostic Procedure

The system recognized the user as a technician and provided a comprehensive, professional-grade response. Instead of a simple guide, it delivered a detailed breakdown as shown in Table 7 covering safety protocols (PPE, ventilation, waste disposal), warranty implications (diagnostic accuracy, genuine parts usage, documentation), and vehicle adaptation requirements.

Table 7. Use Case: Technician—Diagnostic Procedure.

Field	Description
Use Case Name	Technician—Diagnostic Procedure
Input Query	"What is the step-by-step procedure for diagnosing a misfire?"
User Type	Technician (Skilled Service Personnel)
PI-Class Metadata	Intrinsic Product: Vehicle components; Intrinsic Information: Descriptive; Extrinsic Information: Warning
Simplified Category	Other
Purpose or Goal	To provide professional-grade diagnostic workflow ensuring safety and compliance.
Primary Source Referenced	Toyota Camry Repair Manual
Selected UML Diagram Type	Activity Diagram

The accompanying Activity Diagram structured this complex information into a professional workflow, guiding the technician through the interconnected stages of safe practice, accurate diagnosis, and compliant repair procedures. This output demonstrates the system's ability to deliver technical, procedural, and compliance-related information suitable for an expert audience.

The system processed content from the Toyota Camry repair manual [7] to generate this response.

7.2. Use Case 2: User—Comfort/Convenience Feature

For this driver query, the system correctly identified the user's need to understand a comfort feature. It generated a simple guide titled "BMW Climate Control: Setting a Departure Time (Easy Guide)," using the analogy of "setting a timer for your car's air conditioner." The response provided a straightforward, step-by-step procedure for using the iDrive screen to schedule cabin pre-conditioning. The accompanying Use Case Diagram effectively visualized the high-level interactions between the Driver and the iDrive System (use cases: "Set Departure Time," "Activate Pre-conditioning," "Check Settings"), perfectly suited for a non-technical user wanting to understand what the system can do, without delving into technical implementation. This response was generated from the BMW 3 Series owner's manual explained further in Table 8 [8].

Table 8. User-Comfort Convenience Feature.

Use Case Name	User—Controls/Convenience Feature
Input Query	“Can you tell me about iDrive?”
User Type	User (Vehicle Drive)
PI-Class Metadata	Intrinsic Product: Climate Control, Intrinsic Information: Procedural, Extrinsic Information: Display
Simplified Category	Controls
Selected UML Diagram Type	Use Case Diagram
Purpose/Goal	To provide a simple, user-friendly explanation of using the iDrive interface for comfort features.
Final Output Description	System-generated easy guide titled ‘BMW Climate Control: Setting a Departure Time (Easy Guide)’

The contrast between these two use cases validates the system’s core premise of role-aware semantic access. For the technician, the system comprehended the need for a comprehensive diagnostic protocol encompassing safety, compliance, and technical steps. For the user, it correctly interpreted the request as a need to operate a comfort feature, selecting a Use Case Diagram to illustrate user–system interactions simply. This demonstrates that the dynamic PI Classification successfully captures user intent and context, enabling it to personalize the depth, focus, and presentation of information. This role-aware personalization fundamentally improves information retrieval by providing technicians with the necessary technical depth and complexity while offering users simplified, actionable instructions for vehicle features, thereby effectively reducing cognitive load for both audiences.

8. Conclusions and Future Work

This paper presented the Smart Manual Assistant, a system that bridges the gap between static automotive manuals and the demand for dynamic semantic information access in Industry 4.0. Our core contribution is a new approach called Dynamic PI-Classification, which uses an LLM to tag user queries with semantic metadata in real time. This then enables a Role-Aware UML Generation system to automatically produce tailored visual diagrams. We demonstrate the viability of this approach through an automated pipeline that integrates state-of-the-art tools from query to response.

An important feature of the system so far implemented is a transaction log that forms the basis of our future work. Each query–response pair is logged, via Zapier, into a Google Sheet, including the timestamp, user query, user type, PI-Class metadata—all four dimensions, plus a simplified category and diagram type—and the response generated.

Building on this, our future work will leverage this dataset for continuous improvement driven by data through several critical analyses:

- **Measuring PI-Classification Accuracy:** This task entails a manual review of a sampled number of logs to calculate the precision and recall of the LLM in assigning correct PI-Class tags and identifying areas for prompt engineering improvement.
- **Analyzing User Query Patterns:** We will identify the most frequent Simplified Categories; for instance, is Maintenance more frequent or Diagnostics? This will be helpful to manual publishers in prioritizing the content.

In addition, correlation analysis will indicate if certain diagram types lead to higher user satisfaction scores or fewer follow-up questions in the evaluation of UML diagram utility.

- **Performing Gap Analysis:** We identify queries that lead to poor responses to pinpoint certain gaps related either to the source manual or the system’s knowledge. Such

advanced statistical analysis can be done by tools like *julius.ai*, which can directly interface with the Google Sheet, turning the operational system into a powerful research tool to optimize intelligent documentation systems.

This study demonstrates the feasibility and effectiveness of integrating Dynamic PI-Classification with role-aware UML generation to transform static automotive manuals into interactive, context-sensitive knowledge systems. By enabling the system to interpret each user query semantically and automatically produce both explanatory text and corresponding UML diagrams, the Smart Manual Assistant meaningfully reduces cognitive load for drivers and technicians while improving accessibility to complex technical documentation. Looking ahead, the integration of analytics represents a key opportunity for advancing this research. Since every query–response transaction is already logged from Chatbase into Google Sheets, this dataset can be connected to statistical analysis platforms such as Python, R, or Tableau to uncover usage patterns, measure classification accuracy, and identify optimization opportunities driven by real user behavior. However, the current workflow still requires users to manually copy and paste generated Python or PlantUML code into external visualization tools, as attempts to directly automate rendering pipelines through Google Colab or embedded visualization services remain incomplete. This limitation introduces a bottleneck for advanced users and highlights the need for a more seamless, end-to-end visualization pipeline. Addressing this gap would not only improve usability but also align with ongoing efforts to refine UML as an intuitive communication method for technical knowledge transfer, as emphasized by MacKinnon and Murphy in their foundational work on UML design for documentation clarity [9].

Initial Observations (Pilot Analysis)

The pilot implementation of the Smart Manual Assistant demonstrated that the system can reliably classify user queries using the dynamic PI-Classification schema and generate role-appropriate responses together with meaningful UML visualizations. Early interactions show clear differentiation in output depth: technicians consistently received detailed diagnostic workflows and component-level diagrams, while everyday drivers were provided simplified, step-by-step guidance and high-level use case diagrams. The logging pipeline also proved effective, with each query automatically recorded in Google Sheets along with metadata and diagram type, confirming the system's ability to support subsequent analytics and iterative refinement. These initial findings validate the feasibility of the approach and highlight the system's potential to meaningfully reduce cognitive load by delivering context-tailored technical information in real time.

9. Future Development of the Smart Manual Assistant

Looking ahead, the Smart Manual Assistant can be further enhanced by increasing the adaptability and scalability of its semantic interpretation layer. A key objective is enabling the system to recognize and align system components, terminology, and structural conventions across a wide range of automobile manuals, supporting a true single-sourcing model in which one semantic engine can adapt to varying manufacturer styles, product architectures, and documentation formats. This would require expanding the system's internal ontology through AI-driven pattern mining, where Chatbase logs are used to learn recurring linguistic forms, subsystem naming patterns, and procedural structures across different manuals. Additionally, future development could focus on augmenting the assistant's output modalities: beyond static diagrams, the system could trigger just-in-time video or animated procedural guidance generated or retrieved dynamically based on the user's query and role. Through integrations with Chatbase's custom action workflows, the LLM could identify when a query is procedural or safety-critical and automatically

call external media repositories or animation generators to deliver step-by-step visual instructions in real time. This multimodal capability would not only reduce cognitive load during complex operations but also extend the Smart Manual Assistant from a text-and-diagram support system into a fully interactive, adaptive learning and assistance platform capable of guiding both drivers and technicians through maintenance, diagnostics, and feature usage with greater clarity and immediacy.

Author Contributions: Conceptualization, M.A. and D.R.; methodology, M.A. and D.R.; software, M.A.; validation, M.A. and D.R.; formal analysis, M.A.; investigation, M.A.; resources, M.A. and D.R.; data curation, M.A.; writing—original draft preparation, M.A.; writing—review and editing, M.A. and D.R.; visualization, M.A.; supervision, D.R.; project administration, D.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is unavailable due to privacy reasons.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ziegler, W. Extending intelligent content delivery in technical communication by semantics: Microdocuments and content services. *SHS Web Conf.* **2020**, *77*, 03009. [CrossRef]
2. Norman, D.A. *The Design of Everyday Things*; Revised and Expanded Edition; Basic Books: New York, NY, USA, 2013.
3. Sweller, J. Cognitive load during problem solving: Effects on learning. *Cogn. Sci.* **1988**, *12*, 257–285. [CrossRef]
4. Hogan, A.; Blomqvist, E.; Cochez, M.; d’Amato, C.; de Melo, G.; Gutiérrez, C.; Hoeksema, J.; Lecue, F.; Molloy, M.R.C.; Möller, K.; et al. Knowledge Graphs. *ACM Comput. Surv.* **2021**, *54*, 1–37. [CrossRef]
5. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Vancouver, Canada, 6–12 December 2020; pp. 1877–1901.
6. OMG (Object Management Group). *Unified Modeling Language (UML) Specification*, Version 2.5.1; OMG: Milford, MA, USA, 2017. Available online: <https://www.omg.org/spec/UML> (accessed on 2 February 2025).
7. Toyota Motor Corporation. *Toyota Camry Repair Manual: Engine Control System, 2007*; Toyota Motor Corporation: Toyota, Japan, 2007.
8. BMW AG. *BMW 3 Series Sedan Owner’s Manual*; Online Edition, Part no. 01 40 2 911 066; BMW AG: Munich, Germany, 2013.
9. MacKinnon, N.; Murphy, S. Designing UML diagrams for technical documentation. In Proceedings of the 21st Annual International Conference on Documentation (SIGDOC ’03), San Antonio, TX, USA, 20–23 October 2003; pp. 105–112. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.