Context Engines: Enhancing AI Systems for Michelin's Business Rules Project

In the development of Michelin's system for extracting business rules from COBOL code and enabling advanced AI interactions, the role of context engines is critical for improving the accuracy, relevance, and personalization of responses. We have examined several open-source context engines, and three stand out for their potential value in our project: Rasa, Apache Stanbol, and OpenDialog.

Rasa: A Leader in Contextual AI

Rasa is a well-established, open-source framework designed for building AI assistants. It excels in natural language understanding (NLU) and dialogue management, which are crucial for handling complex, multi-turn interactions with Michelin's clients and internal systems. What makes Rasa particularly attractive is its ability to:

- Track and Maintain Conversation Context: Rasa's dialogue management capabilities ensure that the system keeps track of user interactions across multiple sessions. This will allow the system to maintain consistency in conversation, crucial for handling queries about business rules, sales, and marketing.
- Customizable Pipelines: Rasa's architecture is highly modular, enabling the addition of custom components to fine-tune how context is handled. For instance, Rasa can be integrated with our AI solution for extracting business rules from COBOL, ensuring that extracted rules are referenced in real-time based on the user's query.
- Open-Source Flexibility: Since Michelin's project requires both adaptability and scalability, Rasa's open-source nature makes it an ideal choice for building a customizable solution that evolves with Michelin's requirements over time.

Incorporating Rasa’s contextual abilities into our system would provide Michelin with a highly flexible and powerful dialogue engine that can handle varied queries about business rules, orders, and customer interactions.

Apache Stanbol: Contextual Knowledge Management

Apache Stanbol is another powerful open-source framework, though it focuses more on semantic content management than conversational AI. Stanbol is designed to enrich content and manage linked data, which could be highly beneficial when managing the vast amount of business rule data extracted from COBOL code. Key strengths of Stanbol include:

- Knowledge Graphs: Stanbol can link extracted business rules to broader knowledge networks, creating a semantic web that connects information about sales, marketing, and invoicing. This enables the system to provide highly contextualized answers based on not just the content of a rule, but how it relates to other business processes.
- Contextual Entity Recognition: It can automatically enrich and annotate documents with relevant entities, making it easier to retrieve information when users ask questions related to specific business rules. This capability is particularly useful when the system needs to find and deliver highly specific information during user interactions.

Stanbol’s knowledge management capabilities could serve as the backbone for handling and structuring Michelin’s business rules and ensuring the system always retrieves relevant, context-rich information.

OpenDialog: Conversational Design Focus

OpenDialog is another excellent option, as it focuses heavily on conversation design and managing complex dialogue flows. Its design flexibility

allows the
creation of systems that can adapt to user input in a highly contextual way,
making it a strong candidate for Michelin's use case. Key aspects include:

- Complex Conversation Flows: OpenDialog excels in managing intricate
conversations that involve multiple steps and dynamic rule application. For a
system that must respond to business rule queries, client interactions, and
internal processes, this flexibility is critical.
- Context Persistence: OpenDialog allows for persistent context across multiple
conversations, ensuring that the AI understands previous interactions and
responds accordingly. This is ideal for scenarios where users need to
interact over time, possibly across different topics, while still requiring
relevant rule-based responses.

OpenDialog's conversational focus makes it well-suited for user-facing interactions where clarity and flow are crucial, particularly when Michelin's
clients or employees need guidance through rule-based processes.

Integrating Context Engines with RAG and LangChain

The ultimate goal is to integrate these context engines with a Retrieval-Augmented Generation (RAG) framework using LangChain. Here's how it
would work:

1. Document Retrieval: The business rules extracted from COBOL would be
stored in a retrievable format (such as a database or knowledge base). RAG
would retrieve relevant documents or rules based on a user's query.
2. Contextual Relevance: The chosen context engine (Rasa, Apache Stanbol, or
OpenDialog) would maintain an understanding of the conversation's context,
refining the retrieval process by making sure the AI looks for documents or
rules that are not only relevant to the immediate query but also aligned
with the broader context of the user's needs.
3. Generation of Responses: Once the context engine has helped RAG retrieve

the correct documents, LangChain would generate a response that is both contextually accurate and helpful to the user. LangChain's ability to manage complex document retrieval and synthesis would work in harmony with the context engine's task of maintaining conversational coherence.

Benefits of Implementing RAG with Context Engines

- Enhanced Accuracy: The combination of a context engine with RAG ensures that the AI retrieves more relevant information, based not just on the query but also on the history of the conversation.
- Coherent Multi-Turn Interactions: A context engine would help maintain a thread of coherence throughout multi-turn conversations, ensuring users don't need to reintroduce topics or details in future queries.
- Business Rule Application: For Michelin's specific use case, combining a context engine with RAG allows for the dynamic application of business rules. When a user queries about an order or an invoicing rule, the system can retrieve and apply the most relevant rule based on previous interactions and user needs.
- Scalability: As Michelin's knowledge base and set of business rules grow, the use of context engines ensures the system continues to scale effectively, handling larger and more complex rule sets without losing contextual awareness.

Conclusion

In summary, by combining Rasa, Apache Stanbol, or OpenDialog with RAG on LangChain, Michelin's AI system will be able to provide highly relevant, context-aware responses that improve decision-making, enhance client satisfaction, and streamline internal processes. Each context engine offers unique advantages, and selecting the right one will depend on the specific focus—whether it's conversation flow, knowledge management, or dialogue tracking.