FESTO

derivo

Oxford Semantic Technologies

Configuring automation solutions with Knowledge graphs

Context

Adopting IOT and industrial automation requires in most cases designing and assembling specialised products and processes from scratch tailored for the specific use case and meeting the constraints of the industry.

Festo, a worldwide leader in automation and a world market leader in technical training and development, with a turnover greater than 3 billion euros, has a catalogue of over 30,000 products in thousands of variants. Festo's customer use these products as components to realize their pneumatic and electric automation solutions.

Festo might be tasked to "deliver the right components which can be assembled into an assembly solution which has a velocity of 1.2 m/s and a max payload of 2 kg" and will solve the problem by searching through its catalogue for components which are compatible with each other and can be composed together to build the assembly solution.

The number of options to choose from explodes combinatorically with the number of components which means for example, that a hypothetical solution involving 5 components, each available in 10 versions, would result in 100,000 possible configurations to choose from.

This case study will show how Festo was able to completely transform the related internal data processes to reduce the time to provide satisfactory specifications from hours to seconds using RDFox. Oxford Semantic Technologies' partner Derivo implemented RDFox in Festo's Semantic Platform.

Configuring automation solutions

The problem

Identifying components products which can be configured into a automation solution such as a drive train, requires that they:

- Physically fit together
- Have consistent electrical characteristics

Secondly, a customer might provide additional constraints related to their budgets, environment specifications or protection classes of their industry.

Each component has an interface code to characterise these compatibilities. The process of testing for compatible components previously involved joining and looping through interface codes managed by multiple relational databases. This process was extremely time and compute intensive because the diversity of the components meant that there was no universal schema.

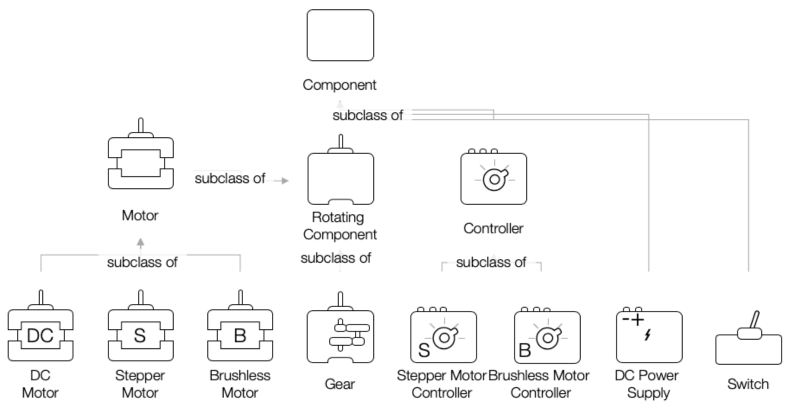
The solution

Derivo proposed an ontological approach by creating a knowledge graph of the solution's components using RDFox.

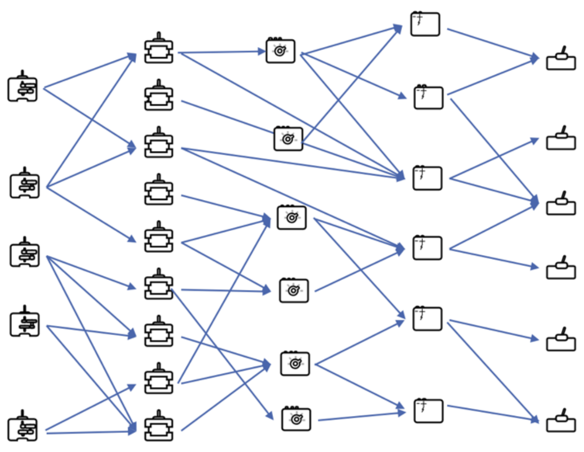
A knowledge graph is composed of a graph database to store the data and a reasoning layer to interpret and manipulate it. Relational databases store data in structured records whereas graph databases store data points as nodes which are connected with edges if they share some form of relationship. Data stored in a graph can be accessed with a query which will "hop" along the edges to find the requested nodes.

Reasoning is the process of materialising rules which apply to the data. Materialising a rule means adding new nodes and edges to the graph when it is satisfied. For example, if a compatibility rule is successful between prudcts, a "compatible with" edge will be established between them. Rules can also encode component families and hierarchies which enables RDFox to "know" how to build a rotation solution and with which components.

A simplified illustration of the types of hierarchies which were encoded is provided in the following graphic.



Compatibility edges are then materialised in the graph by the rules when components are compatible with each other. The result is a compatibility graph that links components which can be assembled together to form specific assembly solutions. A simplified illustration of a compatibility graph for a rotation solution is provided bellow.



When a query is formulated to find the compatible components which can be assembled into a rotation solution it will only have to search for the rotation components form the knowledge graph with materialised compatibility edges in between.

Without rules, the compatibility constraints would have been formulated in the query. This would mean that the query would have to find each rotation solution path and then test for compatibility which is significantly slower and less efficient than going straight to the compatible paths.

Apart from the obvious speed, simplicity and maintainability benefits of RDFox, Festo and Derivo particularly appreciated its flexibility because their customers' requirements are constantly evolving and it is possible to effortlessly add new rules to define the new budget, environmental and domain specific constraints imposed by the customers.

Conclusion

RDFox was able to bring Festo's querying time from hours to seconds by leveraging its powerful reasoning engine. Not only did the rules make it easier to model the problem's constraints, but they materialised the compatibilities before query time which helped deliver significant performance improvements. This approach and RDFox's flexibility also mean that it is easier for Derivo and Festo to maintain and add new rules to reflect their customers' constantly changing requirements.

This case study has highlighted RDFox's capabilities for configuration type problems. It has also been successfully been applied to unlock performance breakthroughs in other applications including fraud detection, risk and compliance, chatbots and recommender engines. For more information, please contact info@oxfordsemantic.tech