



CPQ Concepts Explained: Go Beyond Rules with Constraint-Based Configuration

Using constraints to teach rather than tell

Using constraints, your configuration system can be taught how your products and solutions are put together rather than simply what works and doesn't work. This dramatically reduces maintenance and provides far more intelligent, flexible and future-proof configuration applications.

When evaluating a potential configuration supplier, it's easy to focus on the end result. After all, the people in your organization who are demanding a better sales tool are often the ones who will use it, not the ones who will create it.

While those factors are important, this can lead to overemphasis on the interface and output features that prioritize how users make selections, adjust and apply pricing rules, as well as how integrations to downstream applications such as ERP systems are accomplished.

There is an inherent risk that one of the most crucial factors may be ignored or given low priority: the creation and maintenance of the logic behind the configurable products. There's not much use in having a wonderful looking and well-integrated configuration system if it takes a prohibitively long time to launch new product lines or make updates to the existing ones.

Benefits of Constraints

At Tacton the word "constraint" describes the way we define logical relationships. We tend to avoid the word "rule" because in our world a rule is nowhere as powerful as a constraint.

What's the difference between a rule and a constraint?

A slight improvement would be to create valid combinations of nut-washers, nut-bolts, washer-bolts and then guide the user as they select each component. But it quickly becomes obvious that there are problems to such a system.

Maintenance can easily become a nightmare. What happens as new washers, bolts, and nuts are made



available and others are removed? What if we constantly change which are available in certain markets? What if the valid combination is based on the application? What happens when certain environments exclude certain materials making otherwise fine solutions invalid?

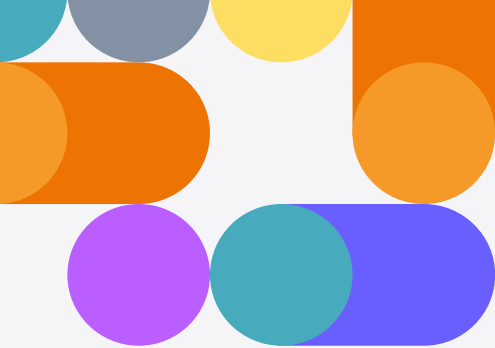
Even with only 100 nuts, 100 bolts, and 100 washers – we have potentially 1,000,000 combinations to manage.

A rule system based on friends (who likes who) and foes (who doesn't like who) can easily become a nightmare to create and maintain. We have seen examples where thousands of old rules are left in the system long after the components they describe are removed from the company's ERP or PDM system.

Sometimes companies reduce their maintenance burden by only creating rules that dictate a certain selection path. First choose the bolt, then the washer, and finally the nut. This means that the only requirement is to create rules that manage those selection transitions.

These rules don't allow the nut to be chosen first since there are no rules covering that path through the product. This then creates a forced path through the configuration application that doesn't allow the user to decide what's important to them. Imagine a chair configurator that forced you to choose material and then color. Once the material is chosen you then see available colors.

But what if you'd prefer to choose a color first and then see available materials. With Tacton that never becomes a problem.



Constraint Logic

Tacton's technology is based on constraint logic. Constraints tend to describe logical conditions that must be met, for instance... "The diameter of the bolt must match the diameter of the washer" "The material of the bolt must match the material of the nut".

To implement such a system, it is often necessary to add characteristics to the components themselves. Perhaps each nut must have specific material, diameter, weight, and cost. Each bolt could have a characteristic detailing in which markets it can be sold, whether it's suitable for deep-sea applications, and with what nut materials the bolt can be combined.

The philosophy is quite simple but highly effective. With such a system we can define constraints that describe each condition. These constraints can be made conditional... "If the application is a deep-sea environment then all materials must be deep-sea compatible". It then becomes possible to apply a layer of constraints on a product that describes market requirements, application requirements, business requirements, as well as guided selling or needs-based logic.

And even better, this constraint-based system doesn't presuppose the order of selection, so users are free to select or answer questions in any order they see fit.

Minimized Maintenance

Over the years Tacton has encountered numerous customers who have suffered the consequences of having poorly designed configuration systems. Even if the creation of simple friend/foe rules is automated, the risk of creating a slow application with hundreds of thousands of rules is very real.

Tacton has encountered situations where a system that

covered only 30% of a product portfolio required 300,000 rules and took half a year to implement. When a Tacton system was used, 100% of the portfolio was covered and it only took one month and 300 constraints. Plus, maintenance and the introduction of new items was significantly easier than before.

In Closing

A constraint-based system:

- Minimizes maintenance by removing the need to write rules when new products are created. Maintenance is required only when logic changes.
- Allow users to follow any path through the configuration.
- Permit the application of different warehouses of options, based on the market to the same system logic.
- Enable fast optimization so that the configurator can quickly find the best solution.