



Driving Change

A giant consumer goods company revamps how it gets products to the store.


A very large consumer products goods company (CPG) had a problem that was core to integrating its different product lines—a common problem in the industry. While the nature of the business around each product line is similar, they grew up as separate entities—essentially separate businesses—each with its own independent infrastructure. Each business would pick up its raw food materials, ship them to factories where finished packaged food goods were produced, ship those to warehouses and distribution centers, and ultimately deliver them to stores. Unfortunately, each business had its own transportation network, so even though their trucks were often visiting the same locations, those trucks were often unnecessarily empty. As they have one of the largest trucking fleets in the world, the wasted trips compounded quickly. As a result, the CPG was leaving a lot of money on the table.

EARLY CONSIDERATIONS

At first, the CPG considered building a new application from scratch using Java. But as the design work began, it became apparent that the complexity would turn this into an ultra-large project that would take a long time and carry a high degree of risk. Each day that went by without a solution had a quantifiable cost to the business, but the cost of a failure in the system would be even more enormous. Without the right kind and quantity of goods in the stores, revenue would take a big hit. So they turned to Ab Initio to solve this problem.

There were many requirements for the new application. It would have to be fully integrated into each of the very large CPG's divisions. It would have to interface with the ordering systems for each. It would have to interpret and process various details of the trucks, factories, warehouses, distribution centers, and stores. It would have to understand aspects of the goods that related to transportation. It would have to enable independent truckers to bid on planned shipments. It would have to connect to all the places where trucks might pick up goods in order to generate bills of lading as well as directions to the crews regarding which goods should be put on which trucks. It would have to feed information to a third-party state-of-the-art optimization engine and communicate the resulting optimized plan to all relevant parts of this global company. In other words, this application would be literally in the middle of everything. And because this business runs 7x24, it would need to be a real-time 7x24 operational system.

RAPID DEVELOPMENT AND A NEW SOLUTION

One would think building such an application for one of the world's largest businesses would require a large effort and a lot of people. But for Ab Initio, that isn't necessary, even on large applications. A single Ab Initio consultant was able to help with architecture, implementation, and all the necessary on-the-job training for a small team of the customer's own application developers, none of whom had used Ab Initio software before. 

When it came to interfacing to the customer's enterprise message bus (in this case, supported by a common messaging product), Ab Initio once again defied expectations. The standard approach is to break down an application into many small programs, each of which reads and writes messages from the bus. All the small programs have to work together in a consistent manner or else the whole will not survive a system failure. Because the responsibility for ensuring robustness lies with the system architects and developers, this often translates into significant amounts of additional coding and testing. And, as is often the case, if the team doesn't anticipate all failure modes, well, you know what happens.

Ab Initio architecture, however, provides robustness features to handle all kinds of failures. As a consequence, the application was reduced to a small number of "graphs" – graphical renditions of the business process and logic. All processes and logic were built entirely in Ab Initio, without resorting to traditional coding. The system was inherently robust because the underlying technology had robustness already built in.

Start to finish, this project was completed in less than one year. And the customer was extremely surprised (and gratified) to discover that once the system was in production, it just worked. Normally, new systems go through a

lot of "teething pain". However, because the application had been substantially simplified with Ab Initio, and because failure handling was built into the underlying technology, the normal challenges melted away.

LOOKING AHEAD

This very large CPG has deployed the new transportation-optimization system across all of its divisions and has reaped tens of millions of dollars in savings. The project's manager, since promoted to SVP, likes to point out that in CIO status meetings, while his peers are bogged down discussing thorny issues with their systems, he gets to talk about his next project, which he is building with Ab Initio, naturally.