White Paper on Technical Software Integration

Arvind Rangarajan
Sponsored by CODEF

Strong theoretical ideas can be embellished by developing accompanying software that can utilize the theory to optimize a process. LMA has been developing burr optimization software following this model to quickly put to use the new theories. Applying the software to real industrial scenarios has become a distinct possibility with the advances made in personal computing and development of geometrical algorithms. Integration has been the bottleneck preventing pervasive application of cutting edge software. The Integration of LMA software with industrial software applications like CATIA V5 or IDEAS has been an arduous task because of the following issues:

- Data required for the software is distributed at various locations in a company's information system. A knowledgeable person who is aware of the location of every single piece of information is tough to find.

- Even if data exists, it might not have the necessary software interfaces to extract them in a very simple fashion.

- Complexity of the optimization tool would mean providing additional information which does not normally exist in a database.

- Testing and quality assurance provided with a software package developed outside is not available in a software tool developed by a university student.

- The concept of "technical support" and "user manuals" does not exist in these software systems.

The thought of integration into a large company wide system can initiate at various stages of software development. The ideal scenario is to identify, plan and execute the elements of integration early. Another commonly encountered circumstance is when the software is developed independently, and its use in a mass production setting is identified after internal verification, which is possible in optimization cases. Despite the precautions taken, the nature of research dictates that new influences will be identified. Hence, having a modular structure will provide the flexibility needed in tuning intertwined software systems. Integration by definition implies two sides of the problem. In subsequent parts, the relevant tasks that need to be worked at to accomplish this at university and industry, are segregated and explained. This procedure can apply for both scenarios with less actual programming in the former case.

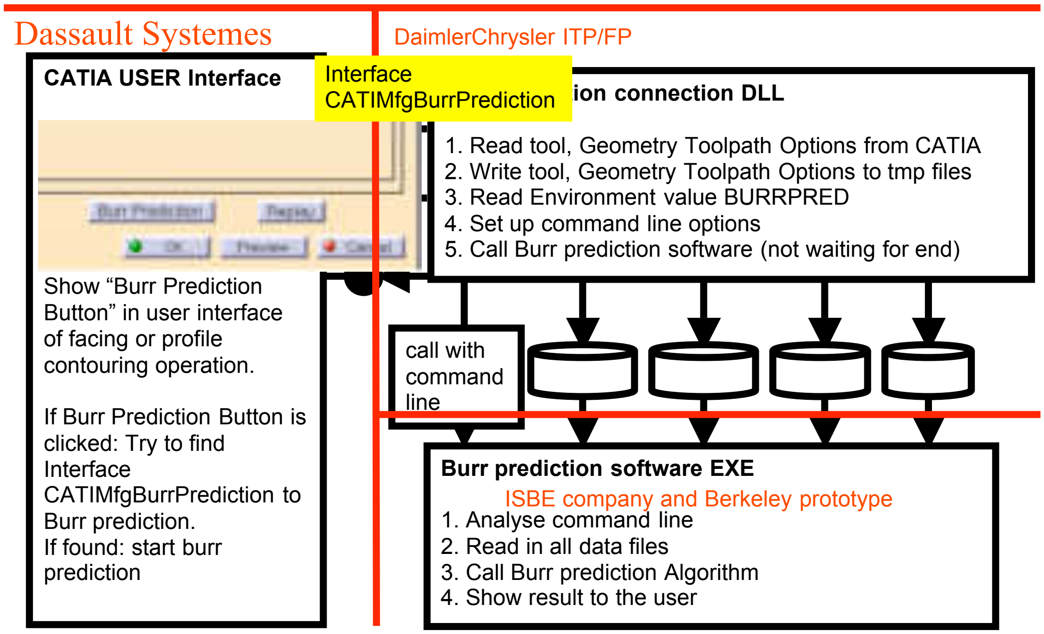
A provision for plugging any optimization or knowledge base tool for a process like milling or drilling is now described. The components of a physical process are generally limited to those that enable, like machine, cutting tools, and those that control the process outcomes like parameter set. The part that is being subjected to a process can also figure in any optimization software tool. Any optimization tool will use information or data from the following superset. A viable solution for the first issue raised is to identify the various databases, existing modules of the product life cycle management system that influence this particular process. The availability of this information in one place provides process relevant information for the optimization tool and gives them location specific details for access.

It can be safely assumed that with the latest software capabilities, one can extract the various interfaces provided in documentation that are related to the particular process, those that enable adding interfaces to the database and other modules to create attributes and add relevant data. A definition for the procedure to add/update attributes to each database or module relevant to the process can be maintained. This implies some operational overhead, but can be very effective for plugging in any new software tool.

The previous two tasks are to be done at the industry and provided to the software developer in a university. There is no real way to enforce the last two stumbling blocks of integration, as the standards of programming at any university cannot be at par with professional software developers. Since the optimization should eventually be applied for the actual parts, a standard part can be assigned as a benchmark for testing with a process optimization tool developed at the university level. This can be independently identified and supplied whenever collaboration is established, preferably in the early stages. This benchmark provides an estimate of the kind of complexity that the software would have to deal with on a regular basis, thereby help establishing reasonable bounds for the algorithm. The user manual and commenting in coding can only be recommended and should be done concurrently with coding. This would help continuation of the work when a new student takes over leading to comprehensive software that can be built upon.

To present an example of integration, Figure 1 demonstrates the steps to be taken for integrating burr prediction software to a CATIA system. In this case, since the interaction with the automotive manufacturer occurred during later stages of software development, it acts as an independent system without feedback. Relevant information is accessed from CATIA and written in a file, which is then read by burr prediction software. The user interface is slightly modified to activate a DLL that calls the steps in the correct sequence, finally starting the burr prediction program. A few interfaces have been added to connect CATIA to external programs.

There are few other minor points that need to be considered, like following version control, high level and low level design documents with frequent checks, to ensure that industry and university people are on the same page. This creates fewer headaches during integration and can hasten significantly the process of incorporating digital optimization schemes in a virtual manufacturing environment. Establishing the above discussed scheme can help accelerate the transfer of software from university to industry and preserve all the functionalities as they were developed.



Show “Burr Prediction Button” in user interface of facing or profile contouring operation.

If Burr Prediction Button is clicked: Try to find Interface CATIMfgBurrPrediction to Burr prediction.
If found: start burr prediction

1. Read tool, Geometry Toolpath Options from CATIA
2. Write tool, Geometry Toolpath Options to tmp files
3. Read Environment value BURRPRED
4. Set up command line options
5. Call Burr prediction software (not waiting for end)

1. Analyse command line
2. Read in all data files
3. Call Burr prediction Algorithm
4. Show result to the user

Figure 1. Roles and tasks in burr prediction software integration.