LINGUAFRANCA - A GRAPHICAL USER INTERFACE FOR ACCELERATOR MODELING

Thomas J. Roberts#, Muons, Inc., Batavia IL 60510, USA

Abstract

This is a proposed project to develop an innovative Graphical User Interface (GUI) that permits users to construct, explore, optimize, and evaluate accelerator systems efficiently and effectively. While it will be designed with students in mind, accelerator physicists will also find it useful in dealing with the plethora of modeling tools and their different languages. The internal representation of the system is specifically designed to be useful as a text-based description of the system, and to make it easy for users to interface it to essentially any accelerator-modeling tool, regardless of its description language. Many accelerator designers have expressed frustration with the current "Tower of Babel" among modeling programs, and this project will address that directly. In particular, this will make it straightforward to use fast but less realistic programs to design and optimize a system, and then use slower but more realistic programs to evaluate its performance. Graphical interfaces are emphasized, making it easy to construct the system graphically, display the system and its beam, and use on-screen controls to vary parameters and observe their effects immediately.

INTRODUCTION

Today's students belong to the "Nintendo generation", and have quite different expectations for educational tools than previous generations. They are accustomed to manipulating graphical representations of objects on computers, tablets, and smartphones, they tend to think visually, and prefer images to text for most applications. In teaching accelerator physics, the existing tools fall short of expectations, and are both cumbersome to use and expensive to purchase. LinguaFranca will address this by providing a free, open-source set of graphical tools for accelerator modeling, integrated into a single package and user interface. Our development team is experienced in user-interface design and implementation, and will create user-friendly tools that interoperate with essentially any existing beam-optics code or particle-simulation program.

As shown in Figure 1, LinguaFranca surrounds multiple existing beam optics and modeling programs with a universal set of tools. The key concepts that permit this to be done both efficiently and effectively are: *commonality* and *instantiation*.

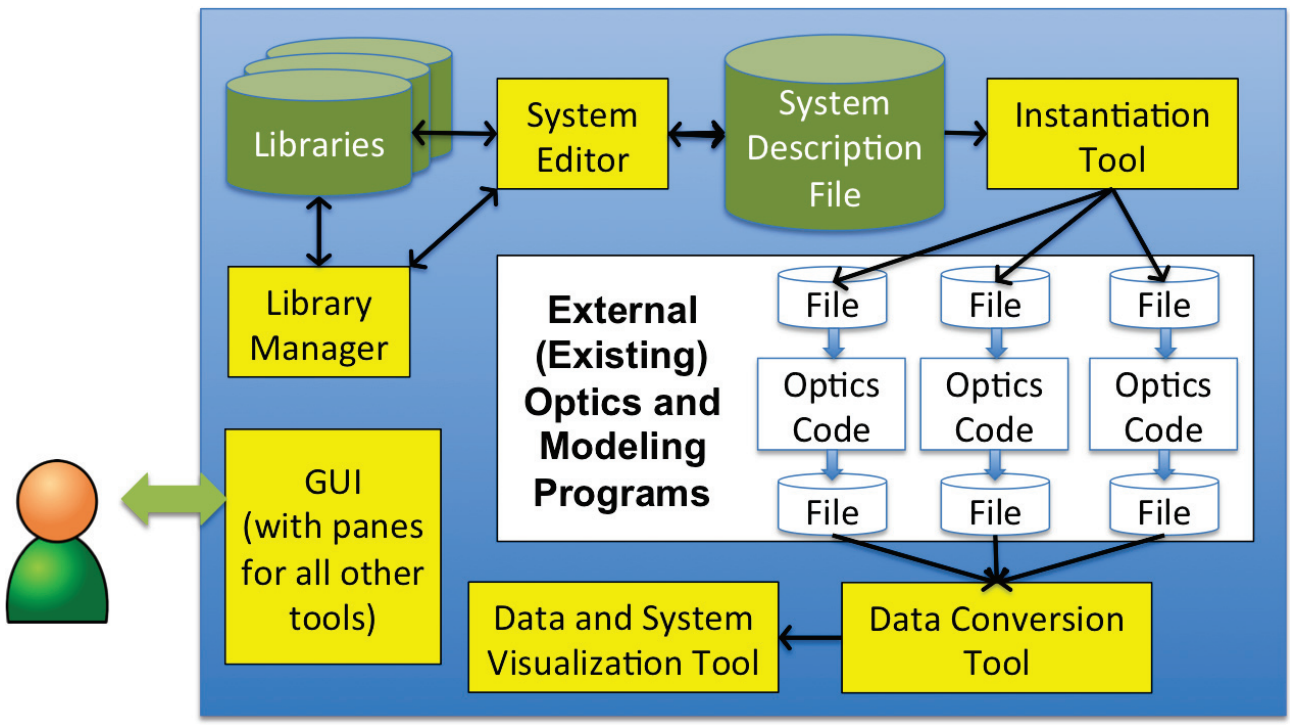

Figure 1: LinguaFranca surrounds multiple existing beam optics and modeling programs with a universal, program-independent set of user-friendly tools.

tjrob@muonsinc.com

ISBN 978-3-95450-115-1

KEY CONCEPT: COMMONALITY

All programs that model or design accelerator systems have considerable *commonality*, in that they all model accelerator systems, inherently using the same basic beamline components (e.g. quadrupole magnets, bending magnets, RF cavities, etc.). Implementing a dozen or so types of components, and permitting them to be placed sequentially on a beam centerline (with optional offsets and rotations), can accommodate the great majority of modern accelerator systems (except cyclotrons and such systems that have no lattice and cannot be handled by lattice design programs). A description of a system in terms of its components will be usable by any modeling program, as long as the tools exist to convert the original description into the input language of the modeling program.

KEY CONCEPT: INSTANTIATION

Rather than relying on language translation, which is a challenging problem even for very simple languages, the approach we use is *instantiation*. That is, a human developer specifies how each component is described in the input deck of each supported modeling program. So a system described as a series of components can be instantiated for any supported modeling program, automatically generating the input deck to run it. The method for doing this is so simple and general that knowledgeable users can easily implement the required instantiations for new components, and for new or unsupported modeling programs.

GENERAL REQUIREMENTS

The LinguaFranca tools will be developed according to these general requirements:

Generality accommodate all types of accelerator and beamline components; support as many lattice design and simulation programs as possible.

Flexibility make it easy for users to move components around, add or delete components, specify component attributes, combine subsystems into systems, etc.

Extensibility permit users to define their own components and subsystems; permit users to add new modeling and simulation programs.

Portability programs must run on most operating system environments, minimally Linux, Windows, and Mac OS X.

Usability implement a modern GUI, but still permit users to easily edit their system descriptions using their favorite text editor.

Sharability make it easy to share components and system descriptions among disparate users via the web; components should be self-documenting.

Explorability make it easy for users to explore the effects of changes, interactively.

Indeed, the LinguaFranca tools need to know very little about accelerators or their components, as that knowledge is encapsulated in the modeling programs. LinguaFranca is primarily a language processing system (the editor and the instantiation tool), plus a GUI, data conversion tools, and display tools.

The primary advantages of this approach are:

- Students (and experienced designers) can "play" with their system, exploring interactively how changes in parameters affect the results;
- Reduced learning curves for experienced designers to use unfamiliar modeling programs;
- Greatly improved ability for designers to select the right modeling program for the job (rather than just using the one they know best);
- For a given system, multiple modeling programs' results can be compared and contrasted much more easily;
- Designers can use graphical tools to construct and visualize the system, even when using modeling programs that don't support it;
- Components are self-documenting and can be published in libraries, fostering collaboration among users and groups via the Internet;
- Considerably reduced development effort compared to either translating description languages or implementing these graphical and comparison capabilities multiple times for the individual modeling programs.

EDUCATION

The primary application of LinguaFranca is to teach students about accelerator physics. For instance, at the elementary level, the effects of a single quadruple or solenoid magnet can be explored. Knowing that a quadrupole magnet focuses in one plane and defocuses in the other is not the same as seeing it on the screen, or watching a movie of the beam evolution through the magnet.

As another example, after having students explore individual accelerator components, the first multi-component system might be a quadrupole triplet focusing to a point; at the elementary to advanced levels the aspects to be taught could include:

- The behavior of beam phase space;
- Manual tuning of magnets;
- Automated tuning of magnets using TRANSPORT (and other programs), and comparison to manual tuning in both ease-of-use and accuracy;
- Exploration of how various initial beam parameters and physical effects affect the ultimate spot size: emittance, momentum variance, dispersion, fringe fields, space charge, wake fields, etc.;
- Comparison to some real beam measurements.

Of course not all of these would be used in any given course. But it's clear that a simple example like this has a wide range of potential lessons, at all levels from elementary to advanced. The intent is that students become familiar with accelerator concepts and the

ISBN 978-3-95450-115-1

behavior of components, while becoming increasingly proficient at using various modeling programs.

GRAPHICAL EDITOR

The graphical editor will be written in Java [1], as will all of the LinguaFranca tools. This gives portability across operating systems, and means that modern tools and advanced libraries can be applied to improve the efficiency and speed of software development.

The basic graphical editing concept is for the user to connect to any desired libraries, and simply drag the necessary components from a library into position in the system being edited. Connected libraries each appear as a window containing their contents, and can contain both individual components and subsystems. Users can also define and use components locally, without putting them into a library.

As our software development proceeds, new modules will be added, to do such things as: tuning and optimizing a beamline, coordinating multiple programs (e.g. tune via TRANSPORT, then evaluate via G4beamline), generating and displaying beam profiles and plots from output files written by simulation programs, plus anything else that seems appropriate. LinguaFranca will become an Integrated Development Environment that streamlines the design and evaluation of accelerator systems.

Figure 2 shows an example of how the graphical editor might look after inserting a quadrupole triplet. Components are represented as icons; for some components like *SimpleQuadrupole*, the icon used depends on the values of attributes (sign of the gradient).

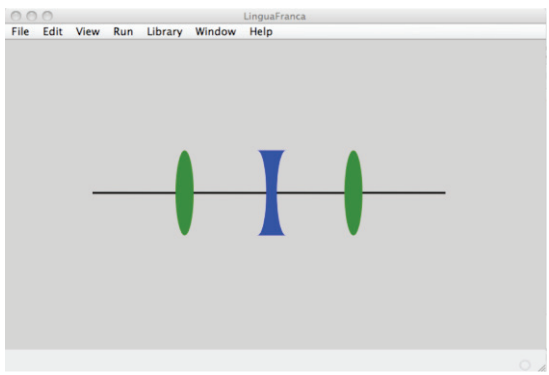

Figure 2: Artist's conception of the Lingua-Franca graphical editor. The three quadrupoles and four drifts were dragged from a library into the system, and their attributes were then edited.

LIBRARIES AND COLLABORATION

Beyond the LinguaFranca base library, it is intended that individual designers, groups, or laboratories publish libraries of their own, providing models of the magnets and other components in their inventory. The design of the LinguaFranca tools and libraries will make it trivial for designers to publish and use libraries containing both subsystem designs and individual components. Self-documenting components are an important aspect of this, avoiding the separation of design data from human-readable descriptions, and permitting complete integration into the tools, including Help. This is intended to foster a new paradigm for accelerator design, modeled after the way modern software is developed, and facilitated by a set of tools that treat such libraries as first-class elements of the language. Rather than just sharing the tools and concepts, accelerator designers will be able to share the results of their design efforts in a simple and flexible way.

For instance, the design effort for a large new facility (such as a muon collider or a linear collider) could be split into numerous teams each working on one subsystem of the overall facility. Each team could publish its own library containing the components it is using in its designs; they could also publish their latest complete subsystem design in a library. An overall team could use the subsystem designs from multiple teams' libraries to verify the matching and interfaces between subsystems; potentially they could combine them all into an end-to-end simulation (although in some cases computational feasibility may prevent this from being useful). It is advantageous to be able to do all this using multiple modeling and simulation programs as checks on each other, under the umbrella of a single user interface.

SUMMARY

LinguaFranca will provide students with modern tools with which to learn accelerator physics, and to explore basic and advanced concepts. It will also give experienced accelerator designers these modern graphical tools, plus the ability to work with multiple modeling and simulation programs much more easily. This, in turn, will assist in the design of large systems by large teams of experts who are geographically dispersed and who may prefer different simulation programs for different parts of the system. By making it easy to compare and contrast the results from multiple simulation programs, this approach can potentially improve the accuracy of such designs, and more importantly, improve the confidence in their accuracy and realism. The incorporation of libraries as first-class language elements will facilitate collaboration in new ways, and will open a new paradigm for accelerator design, using libraries of components and subsystem designs in new designs and systems, rather than starting essentially from scratch each time.

ACKNOWLEDGMENT

Prof. Paul Guèye of Hampton University has made useful suggestions, especially related to education.

REFERENCES

[1] Java, http://www.oracle.com/technetwork/java

ISBN 978-3-95450-115-1