SOFTWARE SUSTAINABILITY AND TOOLKITS:
HOW TO PROGRAM FOR THE ACCELERATOR SIMULATION PROBLEMS OF TOMORROW

David Sagan

Cornell Laboratory for Accelerator Based Sciences and Education
Linac and Storage Ring Programs No Longer Used:

- AGS
- ALIGN
- COMFORT
- DESIGN
- DIMAD
- GUINEA-PIG
- HARMON
- LEGO
- LIAR
- MAGIC
- MARYLIE
- PATRICIA
- PETROS
- PLACET
- RACETRACK
- SYNCH
- TRACY
- TRANSPORT
- TURTLE
- UAL

How many man-years of effort went into these programs?
How can we, as an accelerator physics community do better?
How can software be developed that is useful and is used year after year?
Outline of the Talk

- Software Sustainability – What is it?
- Software Toolkits – What is a toolkit? How do toolkits help sustainability?
- Bmad – An example of a toolkit.
SOFTWARE SUSTAINABILITY
What is Software Sustainability?

A definition of Software Sustainability:

"the capacity of the software to endure. In other words, sustainability means that the software will continue to be available in the future, on new platforms, meeting new needs."

From: Daviel Katz
What Goes Into Software Sustainability?

- **Intrinsic**: The characteristics of the software itself.
- **Extrinsic**: The environment in which the software is developed and/or used.
And there exists Standards for Code Quality

Software Evaluation: Criteria-based Assessment

Mike Jackson, Steve Crouch and Rob Baxter

Criteria-based assessment is a quantitative assessment of the software in terms of sustainability, maintainability, and usability. This can inform high-level decisions on specific areas for software improvement.

A criteria-based assessment gives a measurement of quality in a number of areas. These areas are derived from ISO/IEC 9126-1 Software engineering — Product quality and include usability, sustainability and maintainability.

The assessment involves checking whether the software, and the project that develops it, conforms
Example of **Non-Sustainable** Code

"Software decays relatively quickly if it is not maintained and this is especially true for software used in research"

Mario Rosado de Souza, Robert Haines, Markel Vigo, Caroline Jay,
"What Makes Research Software Sustainable? An Interview Study With Research Software Engineers."
SOFTWARE TOOLKITS
Problem: Developing simulation code takes time.

In general, accelerator simulation problems have some common needs like reading in lattice information or calculating orbits. Writing code from scratch to do this is like reinventing the wheel.

How can we do better?

What does it take to create programs efficiently?
What is a Software Toolkit?

Definition of a Toolkit:

An integrated set of modular software routines that are used to develop and maintain applications or databases.
A Toolkit is like a bunch of Lego blocks

Advantages of a toolkit:
• Cuts down on the time needed to develop programs.
• Cuts down on programming errors (via module reuse).
• Standardizes sharing of lattice information between programs.
• Increased safety: Modular code provides a firewall. For example, a buggy module introduced into the toolkit will not affect programs that do not use it.

A toolkit, by making the software more useful, helps make the software sustainable.
## Accelerator Simulation Toolkits

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accelerator Toolbox</strong></td>
<td>Matlab based. Used extensively at light sources for such things as orbit response matrix analysis.</td>
</tr>
<tr>
<td><strong>Bmad</strong></td>
<td>Includes modules for spin tracking, low energy space charge, etc. Has interface to PTC.</td>
</tr>
<tr>
<td><strong>COSY INFINITY</strong></td>
<td>Uses Differential Algebra (DA) for tracking.</td>
</tr>
<tr>
<td><strong>MAD-NG</strong></td>
<td>In development by Laurent Denieu (CERN). Replacement for MAD-X.</td>
</tr>
<tr>
<td><strong>Merlin/Merlin++</strong></td>
<td>Recent developments oriented towards simulating wakefields and particle/matter interactions.</td>
</tr>
<tr>
<td><strong>FPP/PTC</strong></td>
<td>Has Differential Algebra (DA), symplectic tracking, power series map, normal form analysis, etc.</td>
</tr>
</tbody>
</table>
In the Beginning...

Brief History:

- Born at Cornell in mid 1990’s
- Started life as modest project: Just wanted to calculate Twiss functions and closed orbits within control system programs.
- Initially Bmad used a subset of the MAD lattice syntax. Hence the name: “Baby MAD” or “Bmad” for short.
And Baby Grows Up...

Currently:
- >100,000 lines of code
- ~1,000 routines
And Bmad can do Much More:

- Lattice design
- X-ray simulations
- Spin tracking
- Wakefields and HOMs
- Beam breakup (BBU) simulations in ERLs
- Intra-beam scattering (IBS) simulations
- Coherent Synchrotron Radiation (CSR)
- Touschek Simulations
- Frequency map analysis
- Dark current tracking
- Etc., etc.
Overview

- Written in object oriented Fortran 2008.
- With certain restrictions, Bmad can be run multi-threaded.
- Lattice files use a MAD like syntax.
- Well documented (Manual is >500 pages).
- Open Source: http://www.classe.cornell.edu/bmad/
- And Bmad is indeed a toolkit:

```fortran
type (lat_struct) lat
call bmad_parser ('lat.bmad', lat)
```
Bmad has a number of features that over the years have proven useful. Among these are:

- **Superposition** – Define overlapping elements.
- **Controller elements** – Elements controlling attributes of other elements.
- **Multiple connected beam lines in one lattice** – Example: Injection line attached to a ring.
- **Multipass lines** – Multiple beams sharing a common line such as the IR region in a dual ring colliding beam machine.
- **Custom elements** and **custom particle tracking**.
- Define **beam chamber walls**.

Example of how a lattice can contain multiple attached beam lines and this can be used to describe complicated machine geometries.
Due to its flexibility, Bmad has been used in a number of programs including:

- **Tao**: General purpose design and simulation.
- **Synrad3d**: 3D tracking of synch photons, including reflections, within the beam chamber walls.
- **BBU**: Beam breakup instability simulations.
- **dark_current_tracker**: Dark current electron simulation.
- **ibs_sim**: Analytic intra-beam scattering (IBS) calculation.
- **touschek_track**: Tracking of Touschek particles.
- **freq_map**: Frequency map analysis.
- **MOGA**: Multi-Objective Genetic Algorithm optimization.
- **Lux**: Photon tracking in X-ray beam lines.
- **etc...**

**Code reuse**: Modules developed for one program can, via Bmad, be used in other programs.
Tao: Tool for Accelerator Optics

- **Problem**: Bmad is not a program so it cannot be used “out of the box.” for simple calculations.

- **Solution**: Develop Tao - a general purpose simulation & design program
  - Twiss and orbit calculations.
  - Nonlinear optimization.
  - Lattice design.
  - Etc.

- **Additionally**: Tao’s object oriented coding makes it relatively easy to extend it.

- **For example**: Can add custom commands to interface Tao with a control system (EG: CBETA ERL).

- **Bmad with Tao** gives the flexibility of a Toolkit with the convenience of a program.

**Example**: Element misalignment in the CBETA ERL followed by a steering correction.

**Example**: Modifying the CESR ring to stay within the existing building.
Synrad3D Program

Synrad3D was created to calculate the initial distribution of photoelectrons for electron cloud studies. It has also been used for studying the efficiency of beam masks.

Simulation Procedure:
1. Calculate beam orbit
2. Emission of photons from the beam
3. 3D tracking of photons to the vacuum chamber wall
4. Scattering or absorption at the chamber wall
5. If not absorbed, further tracking until absorption
Synrad3D Development

At the start of development, Bmad provided code for:
- Lattice parsing
- Calculation of the closed orbit
- Photon generation
- Defining the vacuum chamber wall

Synrad3d development involved creating code for:
- Tracking of photons.
- Reflection/absorption at the chamber wall.

→ The use of Bmad saved considerable development time.

Also: The code for defining the vacuum chamber wall came from dark electron simulation development.

→ Code developed for one purpose can be used in other places saving development time.
Synrad3d: SuperKEKB Simulation

Simulations by Jim Crittenden
SuperKEKB wall profile courtesy Takuya Ishibashi

Problem: Photoelectrons can be trapped in the field of a quadrupole. The electrons will react with the beam and if the density is high enough, this can cause problems.

Simulation to determine the number of photons absorbed near a particular quadrupole
Problem: SuperKEKB lattices have a chicane with $p_x \sim 0.03$.

But MAD8/X uses the paraxial approximation which assumes that $p_x, p_y \ll 1$.

So in this case a MAD drift element is a poor approximation for the chicane drifts.
Translation Solution

Translation is done in two steps:
1. SAD -> Bmad
2. Bmad -> MAD

In the Bmad -> MAD step, Bmad uses Etienne’s PTC Toolkit to construct a 2nd order map for the chicane drifts which is then instantiated as a MAD matrix element.

Since Bmad and PTC are toolkits, the translation code is integrated into a unified program.
G-2 Simulation Program

- Dave Rubin at Cornell has been developing a simulation program to simulate the **Muon g-2 experiment** at Fermilab.

- Need to track the polarized muons with:
  - *Injection line into a storage ring.*
  - *Three dimensional field of the injection line.*
  - *Scattering of muons as they cross the inflector wall*
  - *Electrostatic quadrupoles*
  - *Muon decay*
  - *Tracking of electron decay product*
G-2 Simulation

- At the start of program development Bmad provided:
  - Ability to define the geometry of the injection line and storage ring.
  - Ability to define the geometry of the inflector wall.
  - Ability to define custom fields for the injection line and the ability to simulate electrostatic quadrupoles.

- Needed to develop for the program:
  - Scattering of muons through the inflector wall.
  - Muon decay.
  - Etc.

→ Bmad reduced the development time for creating the program and provides a flexible framework for future program modifications.
CONCLUSIONS
Some Thoughts

Main advantage of a software toolkit: The flexibility so that programs can be developed in less time and with fewer bugs.

Main disadvantages of a toolkit: Increased work for the toolkit developers in terms of documentation and stricter code quality requirements.

Personal opinion: The advantages of a toolkit far outweigh the disadvantages for accelerator simulation software.

In particular, Bmad has been successful due to it’s modular, object-oriented design which allows it to be adapted to ever changing simulation needs. The quality and quantity of Bmad based programs would not be possible if Bmad where not a toolkit.

Bmad has been used at Cornell, CERN, KEK, DESY, ANL, BNL, Cockcroft Institute, for simulating CESR, ILC, LCLS2, SuperKEK-B, CBETA, ALS-U, Fermilab g-2 muon project, etc.

Bmad is always in continual development with about 1.5 FTEs. Supported by Cornell Laboratory for Accelerator-based Sciences and Education (CLASSE).

Bmad is evolving towards being able to do “complete” start-to-end simulations.
Thanks to Contributors

Dan Abell
Ivan Bazarov
Moritz Beckmann
Martin Berz
Joel Brock
Desmond Barber
Sarah Buchan
Avishek Chatterjee
Jing Yee Chee
Christie Chiu
Joseph Choi
Robert Cope
Jim Crittenden

Attilio De Falco
Gerry Dugan
Michael Ehrlichman
Ken Finkelstein

Etienne Forest
Mike Forster

Hans Grote
Richard Helms
Georg Hoffstaetter
Chris Mayes

Karthik Narayan
Katsunobu Oide
Mark Palmer

Tia Plautz
Matt Randazzo
Matt Rendina

Dave Rubin
Michael Saelim
Frank Schmidt

Jim Shanks
Jeff Smith
Jeremy Urban
Mark Woodley
Demin Zhou

31
David Sagan KEK September 18-19, 2019
END
For programmers, to maximize Bmad’s usefulness, Bmad is designed to be modular and object oriented from the ground up:

```plaintext
type (lat_struct) lat
call bmad_parser (‘lat.bmad’, lat)
```
Bmad with Tao gives the flexibility of a Toolkit with the convenience of a program.

- **Example:** Element misalignment in CBETA followed by steering correction.

- **Example:** Designing or modifying a machine to be/stay within an existing building.