The U.S./IAEA Workshop on Software Sustainability for Safeguards Instrumentation

Louise G. Worrall, Chris A. Pickett, Oak Ridge National Laboratory
Susan E. Pepper, Katherine M. Bachner, Al Queirolo, Brookhaven National Laboratory

July 2014

Notice: This manuscript has been authored by employees of Brookhaven Science Associates, LLC under Contract No. DE-AC02-98CH10886 with the U.S. Department of Energy. The publisher by accepting the manuscript for publication acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.
DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party’s use or the results of such use of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.
Workshop Objectives
The United States and the International Atomic Energy Agency (IAEA) convened a workshop on Software Sustainability for Safeguards Instrumentation in Vienna, Austria, May 6-8, 2014. The primary objective of the workshop was to assemble a cross-section of all safeguards instrumentation software stakeholders (i.e., users, developers, vendors, and sponsors) to identify strategies for ensuring that critical safeguards instrumentation software products continue to be available for use by the IAEA as required and that software functionality does not degrade over time. Safeguards instrumentation software must be sustained in a changing environment with increasing requirements and limited resources. The approaches taken in the past may not be the best model for the future and, therefore, the organizers wanted to evaluate these past approaches.

Workshop Highlights
Neil Chue Hong, Founding Director of the United Kingdom Software Sustainability Institute, presented the keynote talk on Scientific Software: Sustainability, Skills and Sociology. His presentation highlighted the fact that scientific software has a lifetime that is considerably longer than the lifetime of the associated computing hardware. Therefore, lifecycle planning models for software must anticipate changes in hardware approximately every 2-5 years. Software requires a significant overhaul approximately every 10 years. In his words, software “rots” over time, and therefore, simply doing nothing is not a viable approach for sustainability. For example, one common misconception is that the correct way to preserve source code is to keep it in a repository, but Mr. Chue Hong noted that even in a repository the software has to be maintained. Further, whether called the “bus factor” (Chue Hong) or “lottery factor” (Alexey Anichenko, IAEA), the number of software developers devoted to the sustainment of a key software product should always be greater than one. The points made in this keynote talk were revisited throughout the breakout sessions, and Mr. Chue Hong was quoted throughout the workshop.

The workshop provided the opportunity for external software developers to meet with IAEA staff developers and other external developers. For some external developers, this was the first time that they had met other external developers working on software for safeguards instrumentation. The workshop also provided the IAEA with the opportunity to promote their RAINSTORM project\(^1\) and its benefits. One of the stated goals of the RAINSTORM project is to standardize remote data retrieval and data security for all future IAEA Safeguards Technical and Scientific Services (SGTS) equipment. This is an important goal that will lead to more uniform and shareable analysis software. Discussion of the On-

\(^1\) RAINSTORM is the IAEA’s user requirements for implementing a remote monitoring interface in new safeguards instrumentation designs.
Line Enrichment Monitor (OLEM) and associated software development highlighted the importance of early and iterative collaboration among stakeholders. The development of the Central RADAR\(^2\) Inspection Support Package (CRISP) jointly by Euratom/DG-ENER and the IAEA was also highlighted by the IAEA as an exemplary model for sharing development effort and resources, and the resulting source code. The CRISP software package offers the promise of providing a way to integrate divergent data sources into a common format, which will enhance the ability of the IAEA to develop data analysis software that is more readily shareable.

Figure 1 is a graph prepared and presented at the workshop by Alain Lebrun, IAEA, to illustrate the status of the IAEA safeguards software that is used for portable non-destructive assay (NDA) instrumentation. The graph characterizes software according to whether the software is safeguards-specific (indicates there may be other user communities) and whether the software is owned by the IAEA or another party (indicates the level of access and/or responsibility the IAEA may have to the code for use and maintenance). The codes that are owned by the IAEA and are safeguards-specific are the codes for which the IAEA can take responsibility. The codes that are safeguards specific but are proprietary are of concern to the IAEA because the IAEA does not have the required access to the source code to perform reviews to ensure the software operates as intended or to make necessary modifications. This graph gave workshop participants a very useful framework for identifying critical safeguards software and could also be an important aid for future software sustainability planning.

Summary of Recommendations from the Workshop Breakout Sessions

The workshop was formatted with the delivery of informative presentations each morning and breakout sessions each afternoon. The workshop breakout sessions were structured around multiple relevant scenarios and case studies prepared with input from the IAEA, and time for expert discussions was provided. The resulting discussions among the participants led to numerous recommendations from the participants for improving the management of safeguards instrumentation software. A summary of the significant recommendations from the workshop is provided below.

It is important to the IAEA to have the in-house capability to address software sustainability issues. In particular, the IAEA wants the independence to be able to make minor modifications to software that do not warrant the time and expense associated with a typical member state support program (MSSP) task. In addition, the IAEA would like the flexibility to apply resources, including those available through the MSSPs, as appropriate. For example, in some cases hiring a cost free expert or a junior professional officer is more appropriate than contracting with a vendor, but not in others.

\(^2\) Remote Acquisition of Data and Review
It is widely recognized that the mission to sustain software is a broad and ongoing challenge that encompasses legacy codes and codes that are not yet written, multiple uses and applications, and multiple stakeholders; therefore, there is no single “one size fits all” solution. A key finding of the workshop was the need to develop lifecycle plans for critical safeguards software. For lifecycle planning, the IAEA must create an inventory of current safeguards instrumentation software. Workshop participants recognized that sustainability does not just mean keeping software in use, but it also means knowing when to take certain software out of service or when it is best to re-write or replace the code (e.g., in the case of legacy software). This recognition led to the recommendation that a code audit be conducted to identify the software packages required by the IAEA to support safeguards instrumentation, their relative prioritization, the users and level of use of these codes (including the user community external to safeguards), the maintenance requirements and who is responsible for maintenance, the current cost of maintenance (i.e., capture the cost data), the availability of developers to work on these codes, who owns these codes, and what needs to be done to sustain them. This code audit should also take into account and capture dependencies between MSSPs. This inventory will promote efficient investment in safeguards software by identifying critical software packages and maintenance needs. It will facilitate a gap analysis and will become the basis for software management and lifecycle planning. It is widely recognized that sustainability will require funding, but allocations should be targeted to those codes that are both in demand and of high priority to the IAEA. A
consolidation and prioritized assessment of the portfolio of codes requiring ongoing support and maintenance resulting from the code audit and periodic assessment of new options could also increase the ability of all stakeholders to sustain them. Proper software archiving methods were also discussed by the keynote speaker and should be considered during the audit.

Human resources are a key consideration of software sustainability and sustainability planning. Stakeholders need to be committed and involved in order to successfully tackle the software sustainability challenge. In simple terms, people must be motivated to sustain safeguards instrumentation software and have good reasons or incentives to do so. Software sustainability and maintenance culture must be an integral part of institutional culture and become a routine way of doing business. It was recommended that a “user champion” initiate, lead, and become the proponent for the code audit and sustaining critical software. Code-focused user groups or working groups were also recommended to “socialize” the code, share best practices, and improve knowledge management. Establishment of these groups is a best practice because the groups increase knowledge and understanding of codes, engage next generation professionals, and thereby enlarge the user community. A user or working group need not be expensive or require government or extra budgetary funding. The workshop demonstrated that significant interest and motivation exist among the stakeholders and that a user or working group(s) for safeguards instrumentation software could be formed with minimal encouragement by the USSP or other sponsor.

When codes are in use and there is no immediate plan for upgrades, the subject matter experts (SMEs) and programmers may be reassigned to other tasks and may not be available to address even minor unplanned modifications. It is necessary for stakeholders to devise a plan for ensuring that these experts are available when needed. Applying software development best practices reduces the risks associated with a “bus factor” of one, i.e., a sole developer, and protects users against the unavailability of the SMEs and programmers. A well-structured and documented computer code with a sole developer could, if necessary, be assumed by a competent programmer immediately. There are a number of widely-used, open-source codes that are good examples of this principle.

The participants encouraged investment in sustaining critical safeguards software and supporting associated training. Funding could come from a single “resource champion” or a number of “resource champions.” Options for software sustainability will vary depending on the owner of the codes, but may include planning and providing for a maintenance budget over the lifetime of the software, using umbrella tasks for maintenance, and negotiating technical support contract arrangements with vendors. Another model is the Radiation Safety Information Computational Center (RSIC) system (https://rsicc.ornl.gov), which provides and manages licenses and leverages multiple programmatic support vehicles along with limited user fees to cover the costs associated with software sustainability. Each of these options would ensure that funds are available to support maintenance activities in the timely manner desired by the IAEA. Improved lifecycle planning and a proactive approach to project management would help to ensure maintenance support over the entire software lifecycle. Lifecycle

---

3 Umbrella tasks are MSSP activities that consolidate a number of small, related activities.
planning should, therefore, take into account the “total cost of ownership” for each software product akin to how vendors support key software products.

Timeliness of support from MSSPs was identified as an area for improvement. Recognizing that the IAEA and MSSP processes ensure efficient and effective use of limited financial resources, the approval processes within both the IAEA and the MSSPs can result in delayed access to technical support from the MSSPs.

Discussions regarding intellectual property (IP) led to a recommendation to assess licensing possibilities. The stakeholders need to understand who owns the IP for each of the safeguards software packages and whether the packages can be shared. While some software codes may not be made available to the IAEA, there may be ways to creatively license the software to meet the needs of the IAEA while addressing the concerns of all stakeholders, including those who own the various pieces of IP. Some IP issues could also be mitigated by determining at the start of development who will hold the software IP at the end of development. Again, this dialogue should happen early in the development process and should become a routine part of any development project.

The IAEA believes the noncommercial nature and the small market impact of IAEA activities obviate or lessen the need for IP protection, and the need for IP protection on safeguards-specific software is not justified (see Figure 1). IAEA representatives proposed the concepts of non-exclusive licenses for noncommercial use and partial IP sharing, which would protect proprietary algorithms while open-sourcing architecture and interfaces.

A software escrow can simplify IP issues when agreed to in the planning phase of a software development project. A software escrow is a legal contract which gives the client access to the software developer’s source code and other proprietary materials if the developer becomes incapable of supporting the software. A neutral third party serves as the escrow agent and provides such services as checking that deposited assets are readable and virus free, confirming that decryption keys for encrypted files are on deposit, providing a complete audit and inventory of your deposit, validating that the development environment can be recreated, testing the functionality of the compiled deposit materials, and confirming functionality of released software.

A phased approach to software development could mitigate some of the challenges, such as lengthy development times (interim software products could be implemented earlier) or a product that does not match the IAEA needs (there would be chances to review the project and make corrections at midpoints in the development). Active participation by the IAEA in software development projects should also be part of the phased approach. Software requirements and applicable standards should be defined at the beginning of the project to avoid changes in scope.

The IAEA, as the end-user, must be an active participant in the software development process. It is not acceptable for the IAEA to contract with a developer and remain uninvolved during development. The IAEA must also be actively involved in developments where the contract is between an MSSP and the
developer. Similarly, there should be IAEA champions to promote sustainability of the different instrumentation software programs. This is a challenge due to the IAEA’s “rotation policy,” which results in many professional staff members leaving the IAEA after seven years. Thus, there should be an institutional commitment to software to ensure that software sustainability can span the rotation of the sustainability champions.

Other recommendations encouraged better software documentation and more complete documentation of software algorithms, which would address a variety of problems, including knowledge management and the ability for software to survive unavailability of the software developer. It was recommended that teams of SMEs and software developers consult with technical writers to produce high quality documentation. In particular, the IAEA could prepare software requirements to document the required functionality for vendors to use in preparing software. The requirements can be updated as new measurement approaches emerge. This approach addresses both the “rot” problem and the IAEA’s desire to have source code and allows the IAEA to define the requirements for the software without having to own it. A system that does not meet the requirements would not be saleable to the IAEA.

Innovative and promising approaches, such as the CRISP joint development and the OLEM instrumentation project, should be benchmarked. It was recommended that more experience should be gleaned from development partnerships or the use of RAINSTORM. Furthermore, success indicators or metrics of future software development projects should be clearly defined for future projects. The safeguards community should learn from other scientific communities that have previously faced and addressed the software sustainability issue.

Specific technical recommendations include the use or improved implementation of modular programming methods, which was regarded by the participants as an essential component of programming. Modules of safeguards software would include data acquisition, data management, and data analysis. This would keep the functional elements, which may be proprietary, separate from the interfaces, which may be customized for the IAEA’s use, and facilitate desired access to the code for the IAEA. Standardization of software features, such as basic modules and input/output formats, was also recommended for the future.

Workshop participants agreed that the IAEA should seek feedback on RAINSTORM. The IAEA has implemented RAINSTORM in several systems including the Universal NDA Data Acquisition Platform (UNAP), the Laser Mapping System for Containment Verification (LMCV), the Next Generation Autonomous Data Acquisition Module (NGAM), OLEM, and other instruments and sees it as a standard for the future. However, because workshop participants were not widely familiar with RAINSTORM prior to the workshop and were only familiar with it through its application in OLEM, they recommended more review. This recommendation supports the establishment of a user group that includes people who are knowledgeable in all aspects of developing and sustaining software. While RAINSTORM is not yet a standard, the workshop participants applauded the IAEA’s initiative in developing this product
which will one day serve that role and encouraged the community to develop other such standards and associated requirements.

Acknowledgements
This project was funded by the National Nuclear Security Administration’s Next Generation Safeguards Initiative’s Safeguards Technology subelement. The Department of State’s High Priority Safeguards Program provided funding to cover the expenses of many of the private sector attendees. The workshop team would like to give special acknowledgement and thanks to Jim Regula (IAEA) who worked closely with the workshop team to plan for, develop, and make arrangements for the workshop, Emil Farkas (IAEA) and Chris Orton (NNSA) for facilitating workshop breakout sessions, David Peranteau (IAEA) and Hilary Lane (NNSA) for taking notes in the breakout sessions, Inna Cherkasskaya (IAEA) for taking care of many of the logistical aspects in Vienna, Barbara Hoffheins and Ben Deering (U.S. Mission to International Organizations in Vienna) for providing on-site assistance in Vienna, and Laura MacArthur and Michele Rabatin (BNL) for providing administrative assistance to the workshop. Finally, the workshop team appreciates the participation and contributions of the many software, hardware, and international safeguards experts who attended the workshop.