INTECH
open science | open minds

OPEN ACCESS
ARTICLE

International Journal of Advanced Robotic Systems

Developing an Interactive Intervention Planner - A Systems Engineering Perspective

Regular Paper

Thomas Fabry[1,*], Liesbeth Vanherpe[1], Bruno Feral[1] and Christian Braesch[2]

1 CERN, Geneva, Switzerland
2 SYMME, Université de Savoie, Polytech Annecy-Chambéry, France
* Corresponding author E-mail: thomas.fabry@cern.ch

Received 1 Mar 2013; Accepted 9 Jul 2013

DOI: 10.5772/56846

Abstract Intervention planning is crucial for maintenance operations in particle accelerator environments with ionizing radiation, during which the radiation dose received by maintenance workers should be reduced to a minimum. In this context, we discuss the development of a new software tool and the entailed methodology, including the visualization aspects. The software tool integrates interactive exploration of a scene depicting an accelerator facility augmented with residual radiation level simulations, with the visualization of intervention data such as the followed trajectory and maintenance tasks. Its conception allows for future inclusion of measurements performed by mobile robotic devices. In this work, we explore the systems engineering life cycle of the development process of an interactive intervention planner, which includes the needs analysis, specification explicitation, conceptual mathematical modelling, iterative implementation, design and prototype testing and usability testing.

Keywords Interactive Data Visualization, Ionizing Radiation, Intervention Planning, Systems Engineering

1. Introduction

The work in this paper is primarily closely related to particle physics experimental areas. Particle physics is a branch of modern physics studying the smallest known constituents of matter. Particle physics research necessitates large and complex scientific instruments: particle accelerators and detectors [1, 2]. The circulation and collisions of high energy beams in the accelerators and detectors have an undesirable consequence, namely the radiological activation of some of the components of accelerator facilities [3].

The complexity of particle accelerators and detectors leads to the frequent necessity of maintenance operations. To protect maintenance personnel from ionizing radiation during interventions in the particle accelerators and detectors, the so-called ALARP or ALARA approach (As Low As Reasonably Possible or Achievable) [4, 5] is mostly used, which consists of justifying, optimizing and limiting the dose received by all those who need to work on activated components. Because of this, a core issue during the planning of a maintenance intervention in a facility with ionizing radiation is the minimization of the dose the maintenance workers are subjected to. This optimization

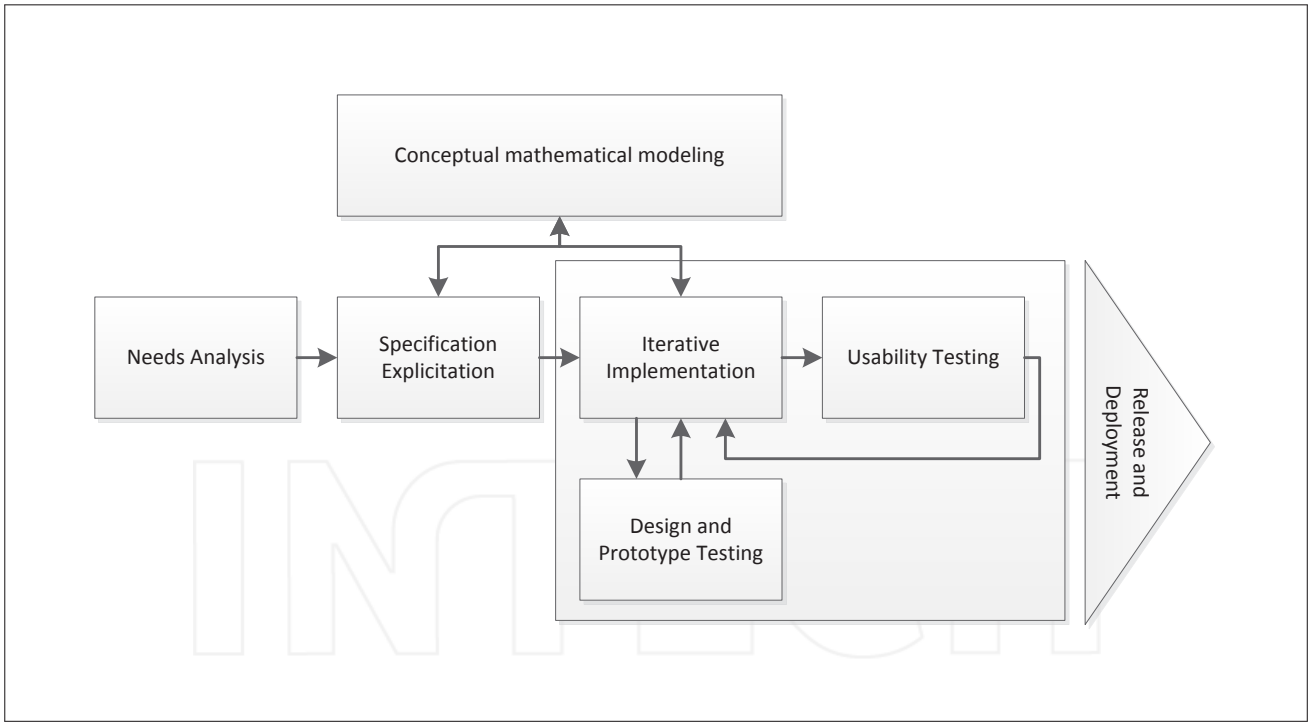

Figure 1. The systems engineering life cycle for the development of an interactive intervention planner.

cannot be automated since the practical feasibility of the intervention tasks requires human assessment, but the intervention planning could be facilitated by using a software tool with three-dimensional visualization capabilities. The development of this tool is a complex undertaking for three reasons. Firstly, the visualization has to cover the infrastructure, the (expected) radiation levels in the facility and the intervention. Secondly, this visualization has to be intuitive to work with for all stakeholders involved (intervention planners, scientists, maintenance workers, safety officers,…) and useful in different scenarios (visual training of operators, three-dimensional visualizations to support the decisions of the ALARA committee,…). Thirdly, the application is about the safety of humans and is therefore not allowed to have any kind of ambiguity. These three points lead to the necessity of a good systems engineering approach.

This article deals with the development process of a methodology and software tool providing interactive visualization for intervention planning in particle accelerator environments with ionizing radiation. In section 2, we discuss the various phases of the systems engineering life cycle: needs analysis & specification explicitation (section 2.1), conceptual mathematical modelling (section 2.2), iterative implementation (section 2.3) and usability testing (section 2.4). Section 3 discusses the resulting application, developed following this systems engineering approach. Section 4 discusses a possible future direction for the development and the systems engineering life cycle. Section 5, finally, concludes the paper.

2. The systems engineering life cycle of the development process of an interactive intervention planner

Systems engineering life cycles are a very important aspect in the accomplishment of a particular objective according to plan [6]. The systems engineering life cycle of the development process of an interactive intervention planner, the analysis and synthesis of the problem parts in the development of the interactive intervention planning application, is shown in Figure 1. The structure of the systems engineering life cycle also allows for clear documentation: every block of the life cycle also includes a documentation phase. The different phases of this research, development, test and evaluation (RDT&E) life cycle are discussed in the following sections.

2.1. Needs analysis & specification explicitation

In every project, be it an Information Technology (IT), construction, industrial, organizatorial change or new service development project, identifying user needs is of key importance for the successful completion of the project [7]. Although this project is a research project and therefore a relaxed systems engineering approach might have to be adapted, it is no exception in that the needs are important to start with. However, identifying user needs is also "the most difficult, most critical, most error prone and most communication-intensive aspect of software development" [8]. In addition, the needs will typically be more easily changed during a research project than during any other project.

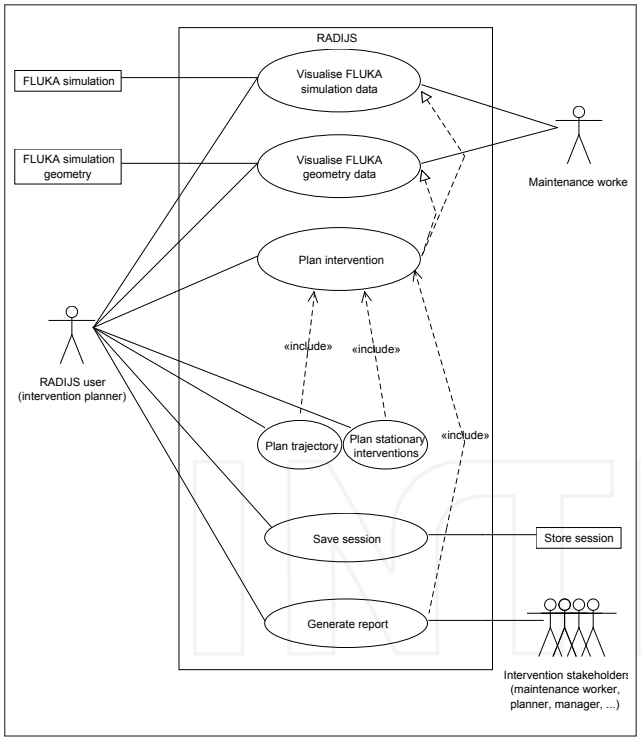

Figure 2. The use case context diagram. This diagram shows how the different stakeholders (depicted as named stick figures) and external systems (depicted as named boxes) are expected to interact with the software system, here indicated as 'RADIJS'.

In addition to this, at the start point of our life cycle, the needs analysis or user needs study is particularly important because of the set-up of this project: the user needs are not centred around one user group. They are distributed around many stakeholders: the intervention planner, the maintenance worker, the radiation protection experts, and almost all persons involved in a particular particle science experiment or equipment.

Because this is a research project and because of the diversity of the user needs, we decided to go for a low-profile way of needs gathering. We did not organize formal *customer panels*, but attended various meetings and discussed in an informal, non-intrusive way the potential applications of the software for visualization of radiation levels with people that are concerned with this type of problem. It became clear that in the current situation, powerful three-dimensional visualization techniques are not consistently used for the visualization of radiation levels at CERN. However, both simulated and measured data from manual measurements and from a fixed survey system are used. In the future, data measured by mobile robots might also become available [9–11].

In addition, because of the diversity of user needs and the research nature of the project, it is extremely important to explicitate the software specification in a way that is as simple and straightforward as possible, while keeping the information content high. This is why we opted for use cases [12] to communicate the specifications. The use cases are based on the gathered user needs that are mapped in Table 1, together with their estimated importance. This table also shows the nature of the input data of the software, which is an important outcome of the needs analysis. The use case context diagram for the developed use cases can be seen in Figure 2. We have explicitized the functional specifications in this way as the use case context diagram is widely recognized as the simplest graphical representation of the interaction of the user with the to-be-developed software. It portrays different types of user-software interactions in a very intuitive way, namely, it shows how the different stakeholders (depicted as named stick figures) and external systems (depicted as named boxes) are expected to interact with the software system, indicated in the figure as 'RADIJS'.

An important outcome of the needs analysis and specification explicitation phase is the starting point of the data flow of our application, i.e., the radiation simulations and three-dimensional geometry. At CERN, the FLUKA Monte Carlo simulation package [13, 14] is used for radiation protection studies, as FLUKA has its roots in this field and is thus the most appropriate choice for these studies [15]. FLUKA is well benchmarked in this area [16–20]. It will thus be necessary for our application to be able to deal with data that is the output of a FLUKA simulation, and with the geometry data that is given as input to FLUKA. In Figure 2, this is expressed by the two uppermost use cases.

Data from manual measurements and/or robotic measurements are to be considered in a further phase of the development. These data will not have the dense nature that the simulation data has, and will thus need interpolation. This interpolation is however far from trivial [21], and much research will be needed to make this feasible. Augmenting the simulated data with measured data, to assess the quality of the simulated data, is more promising (see section 4).

2.2. Conceptual mathematical modelling

As the intervention planning software will be used in a scientific environment and, more importantly, will be used to assess the safety of maintenance workers, a rigorous mathematical model of the intervention planning is necessary. This modelling includes various planning concepts, such as the intervention $\mathcal{I}$, a trajectory $\mathcal{T}$, and the contracted radiation dose H.

An intervention $\mathcal{I}$ consists of a set of tasks T_k, each with a specified description and duration:

$$\mathcal{I} = \{T_k; \quad k = 0, 1, \ldots, K\}. \tag{1}$$

These tasks are parts of the intervention that has to be performed, starting with the entrance of the facility by the worker (T_0), and ending with the worker exiting the facility (T_k).

A trajectory $\mathcal{T}$ consists of a series of locations m_i joined by paths S_i, with $i = 0, \ldots, N$. Each location and each path can be associated with certain radiological

properties, that can be deduced from the radiation dose rates that are available from the FLUKA simulations. The equivalent dose H received by the maintenance worker performing an intervention $\mathcal{I}$, which is mapped on a trajectory $\mathcal{T}$, is then calculated as the sum of the radiation received at the specified locations m_i and the radiation received over the paths S_i between the locations, which the maintenance worker travels along with a velocity v_i:

$$H(\mathcal{I},\mathcal{T}) = \sum_{i=0}^{N} t_i \dot{H}(m_i) + \sum_{i=0}^{N-1} \int_{m_i}^{m_{i+1}} v_i^{-1} \dot{H}(s)\mathrm{d}s, \quad (2)$$

where s is a point on the path S_i. The radiation rates $\dot{H}$ are available from simulations of the activation of facility equipment, or could be available from manual measurements performed in the irradiated facility. For more information on the mathematical model, we refer to [21].

The model as described above is able to deal with manual measurements as well as measurements collected by a robot. While the interactive intervention planner is intended for planning interventions where the work cannot be done by a remotely operated vehicle, it is imaginable that it is possible for a robot to perform a pre-inspection task, of which a validation of the simulation used for the intervention planning can be an outcome. Efforts on such mobile robotic devices are underway in this context [9–11].

To make this model as useful as possible, the conceptual mathematical modelling effort is developed in parallel with the specification explicitation and iterative implementation phases, as can be seen from Figure 1, and is being published in order to be checked by the wider scientific community. The mathematical modelling is compliant with the intervention planning needs at stake, with radiation protection theory [5], and sound to be implemented.

2.3. Iterative implementation & design and prototype testing

Iterative software development methods are used by many organizations to reduce development risks and to deliver software projects on time [22, 23]. Design and prototype testing are integral parts of the iterative implementation strategy. In addition, this software development project makes use of an iterative implementation method.

To facilitate fast development, we opt to develop in Python [24]. Python is a general-purpose, high-level programming language whose design philosophy emphasizes code readability. This is a very important programming language quality in the collaborative context at CERN. Python supports the object-oriented programming paradigm, which is needed for a project like this. While Python is often used as a scripting language, we thus use it in a non-scripting context. Another important factor is that, using third-party tools, Python code can be very easily packaged into stand-alone executable programs, and that Python interpreters are available for many operating systems.

	User need	**Estimated importance**
1.	Intuitive visualization	⋆⋆⋆
1.1.	CAD-like visualization of geometry	⋆⋆⋆
2.	"Easy-to-read" visualization	⋆⋆⋆
2.1.	3D visualization	⋆⋆⋆
2.2.	Interactive visualization	⋆⋆⋆
3.	Easy-to-use software	⋆⋆⋆
3.1.	Intuitive interaction possibilities	⋆⋆⋆
3.2.	Intuitive GUI	⋆⋆⋆
3.3.	Usable on normal PC hardware	⋆⋆
3.4.	Easily installable	⋆⋆
3.5.	Cross-platform	⋆
4.	3D interaction possibilities	⋆⋆⋆
4.1.	3D on/off interaction possibilities	⋆⋆
4.2.	3D camera interaction possibilities	⋆⋆⋆
4.2.1.	Free movements of camera	⋆⋆⋆
4.2.2.	Camera zoom	⋆⋆⋆
4.3.	3D labels	⋆
5.	Possibility to save program status/scenarios	⋆
6.	Possibility to export 2D images	⋆
7.	Possibility to import simulation data	⋆⋆⋆
7.1.	Possibility to import from FLUKA	⋆⋆⋆
8.	Possibility to import geometry	⋆⋆⋆
8.1.	Possibility to import a 3D file format	⋆⋆⋆
9.	Possibility to import measured data	⋆
10.	Possibility to input various scenarios	⋆⋆⋆
10.1.	Possibility to input trajectories	⋆⋆⋆
10.2.	Possibility to input trajectory properties, such as moving speed	⋆⋆⋆
10.3.	Radiological calculations	⋆⋆⋆

Table 1. Needs table and importance mapping.

For the visualization library, the Visualisation ToolKit (VTK) [25, 26] was selected. VTK is a well-known, open-source and freely available software system for three-dimensional computer graphics, image processing and visualization. VTK consists of a C++ class library and includes a Python interface layer, is cross-platform and runs on Linux, Windows, Mac and Unix platforms.

For the development of the graphical user interface (GUI), we choose to make use of wxPython [27]. Because major attention has to be paid to the requirement of an intuitive graphical user interface allowing fast and flexible visualization, trajectory creation and reporting, the user interface (UI) is as much as possible decoupled from the back-end of the software [28].

During this phase of the software development, many implementation iterations were run through. Each time, a prototype version of the software was tested by several users. The resulting prototype test results were used as an input for a new iteration of implementation. This phase of prototype testing distinguishes itself from the phase of usability testing described in the next section, in that intermediate prototype versions of the software were tested for practical reasons, i.e., the correct functioning of

the software, such as successfully loading data and the utility of interaction tools.

2.4. Usability testing

The central idea of the usability testing (top right in Figure 1) is to test whether the software arrives at the intended result and to determine the optimal settings for the software to be user-friendly for as many of the stakeholders as possible. The needs table that was developed during the needs analysis and the specification explicitation phase, as discussed in Section 2.1 and more specifically Table 1, is therefore the guide.

Secondarily, since the interactive and three-dimensional visualization tool for the planning of interventions in facilities emitting ionizing radiation is not implemented yet in the facilities it has been designed for, usability tests are needed to prove that the application of these techniques are indeed useful to intervention planners.

The usability tests are split into two phases. The main goals of the first phase are, firstly, to qualitatively prove the usefulness of the three-dimensional visualization for the user, and secondly, to make way for larger usability tests using more quantitative variables in order to discover the optimal settings for the three-dimensional visualization. More information on these first-phase tests can be found in [29]. In a second phase, more extensive tests are pursued to make way for the release and deployment of the application.

For this second phase, we propose developing a test where a large number of users each go through the intervention planning process of a real-life situation at CERN, for different well-known, existing visualization methods. The test users will originate from all stakeholders involved in intervention planning. The results of the intervention planning, such as the simulated contracted radiation dose, will be studied to obtain visualization parameters that are optimal for the application. Furthermore, the subjective feelings of the user with respect to the visualization will be examined. At the same time, the user will be questioned on the planning experience to assess whether the needs listed in Table 1 have been met.

This recording of the subjective feelings of the test subjects will be done in an informal way, by having an informal chat with the test subject after the usability test. In this way, a very coarse retrospective analysis of the performance of the software tool's visualizations is envisaged. However, no formal think-aloud protocol (when the subjects are taking part in the usability test they think aloud as they perform the tests) will be implemented. A concurrent think-aloud protocol would make the timings that will be recorded less reliable, as is proven in [30], while a retrospective think-aloud protocol would make the usability testing infeasible due to time constraints. In addition to this, the methodological foundations of think-aloud usability testing are still questioned with regard to scientific value [31].

The most distinctive part of the usability testing in this particular project, the study of the optimal volume rendering parameters, will consist of a thorough investigation of the influence of the values of the volume rendering parameters that we presuppose to be important for the acceptance, the usability and the usefulness of the software in the context of CERN operations. To our knowledge, similar previous studies were always limited to:

- academic examples [32–34],
- very well-defined visualization or interpretation subtasks of visual data analysis [33, 35],
- static images [32–35] and
- specific, very specialized rendering methods or environments [32–36].

We propose developing an interactive user study of a real-life situation at CERN, using well-known, existing rendering methods. The planned second-phase usability tests will therefore be more extensive and their results will be compared to the more specific studies in the literature. We thus aim to demonstrate the usefulness of volume rendering techniques and visual data analysis for the empirical science of radiation protection.

That even small changes in the volume rendering technique can have significant effect, and what kind of effect they lead to in the visualization can, for instance, be appreciated from the figures in [35].

In the spirit of systems engineering, usability tests are an important step in the project and can lead to valuable insights in the iterative development process. Because of this, and because of the specific nature of software development, we want to proceed to a usability test as soon as possible.

3. The resulting application

The core of the resulting application is the visualization capacity for FLUKA simulation results and the geometry that comes with it. Due to the nature of FLUKA simulation data and the requirement of a clear visualization of the working conditions, volume rendering is the natural choice for visualizing the radiation levels augmented on the facility geometry. As we want to be able not only to see the radiation levels on certain positions in the three-dimensional space of the facility, but also inside the volume that makes up the facility, volume rendering is the only feasible choice. We consider volume rendering to be a very intuitive volume visualization technique, compared to, for example, volume slicing. Volume rendering has been around for many years [37, 38] and recently the development and improvement of off-the-shelve GPUs has led to the proposition of several interactive advanced volumetric illumination models [39].

Architecturally, the application consists of two main packages, and a number of supporting modules. It makes use of a number of well-known design patterns, such as the Façade, Observer and Iterator patterns [28].

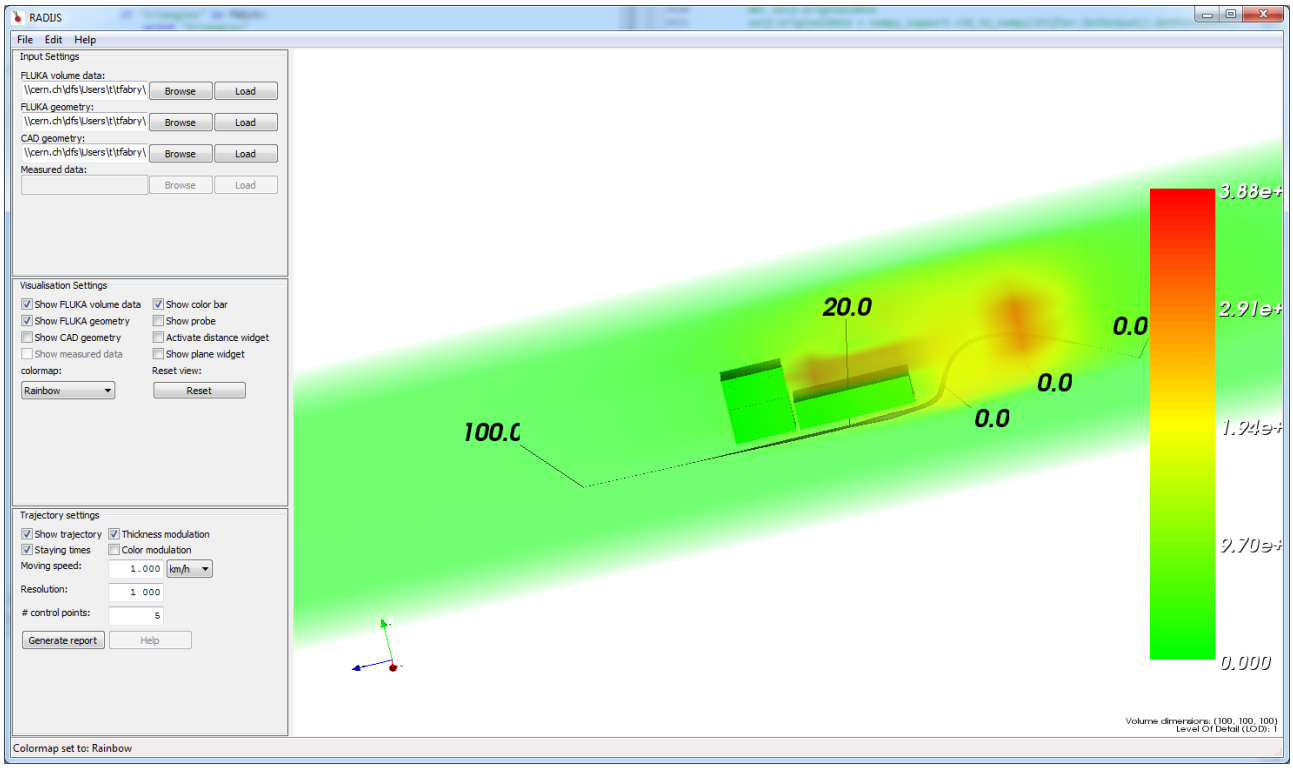

Figure 3. The user interface of the visual intervention planner.

The two main packages are a framework package for the processing of the facility geometry and radiation (simulation) data, and a GUI (Graphical User Interface) package. In the context of this paper the architecture will not be fully discussed, but it suffices to point out that by using an iterative development methodology, embedded in a rigorous systems engineering life cycle, an elegant design can be obtained.

Arguably one of the most important aspects of the software, certainly in the context of this particular software project and as outlined before, is the user interface. A screenshot of this interface, the GUI of the resulting application, can be seen in Figure 3.

The application is made so that it is intuitively possible for every stakeholder in the intervention planning process (intervention planners, scientists, maintenance workers,...) to assess the important features of the intervention. This means that for every possible user of the software, with his or her own personal background and interest in, e.g., radiation protection, practical implication of certain technical interventions, transport requirements, ..., it has to be possible to *see* the variables he or she is interested in being visualized by the software. It is thus possible to make a good *assessment of the conditions in the facility*, by investigation of both the geometry of the facility and the volume-rendered (simulated) radiation levels. The visualization is interactive and allows zooming or panning to gain a better view. In addition, there are tools which enable a closer look at the radiation levels at specific points.

Next, it is possible to *prepare a trajectory* in the facility and map the tasks to specific locations of the trajectory. To allow for *optimization of the intervention*, the software provides tools to add locations to the trajectory, refine and edit the trajectory, and move existing locations.... At any time, it is possible to generate a *report* showing the radiological impact the intervention will have on the persons involved in the intervention.

The GUI is very simple in conception. The application interface is divided in two regions: a region for the interactive visualization and a region for the various settings. This latter region is divided into three boxes for, respectively, input settings, visualization settings and trajectory settings. The settings are preset to values that have been empirically proven to be meaningful for the CERN cases that we have been provided with as test cases.

4. A possible future direction: robotics integration

So far we have described a systems engineering life cycle for the development of an interactive intervention planner with human and computer actors. In the future, there may however also be a need to include robotic actors. In this context, we discuss a particular use case that could be part of an extended systems engineering life cycle for the development of the intervention planning methodology in general, namely, the validation of the latter methodology using programmable mobile radiation-measuring robots.

Until now the software has relied on FLUKA simulation data for its operation. This is justified because FLUKA was extensively validated for use in radiation protection

around high-energy accelerators [16–20]. By integrating the software with a mobile robot equipped for radiation detection, which is under development [9–11]], the validation of individual intervention scenarios constructed with the software tool will become feasible.

With the availability of a radiation-detecting mobile robot, a use case can be envisaged where the trajectory generated with the intervention planning software is used as the input for the programming of a trajectory of the robotic device. The robot shall therefore be equipped with a suitable radiation sensor, so that it can measure radiation levels while covering the trajectory. With the results of the measurements taken by the robot, both the FLUKA simulation data and the interactive intervention planner can be validated in a fine-grained way, taking into account all (possibly hidden) variables that come into play when planning the intervention. This will further strengthen the validation of the simulations or, alternatively, provide new input data for strengthening the simulation code. If robotic devices become more powerful and - for some interventions - suitable to replace human maintenance workers, similar use cases can be imagined to plan robotic interventions.

5. Conclusion

Particle accelerators and detectors used in particle physics research can lead to ionizing radiation and their components becoming activated. This in turn leads to facilities where ionizing radiation is present. To protect the accelerator facility maintenance personnel from ionizing radiation during maintenance or repair interventions, the radiation dose received by the workers during an intervention has to be optimized.

In this work, we outlined the systems engineering life cycle of the development of a software tool for interactive planning of interventions in environments with ionizing radiation. This development is a complex problem with many aspects. The different steps of the systems engineering life cycle were discussed, including a needs analysis, specification explicitation, conceptual mathematical modelling, iterative implementation, design and prototype testing and usability testing. The result of this rigorous approach is a well-documented and purposeful software tool with demonstrated potential.

This work contributes to the important question of the feasibility of adapting a (relaxed) systems engineering approach in rather complex multi-disciplinary research projects.

6. Acknowledgements

This research project has been supported by a Marie Curie Fellowship of the European Community's Seventh Framework Programme under contract number (PITN-GA-2010-264336-PURESAFE).

The authors would like to thank Chris Theis, Keith Kershaw and Pierre Bonnal for the valuable discussions, as well as Mathieu Baudin for the tedious task of beta-testing.

7. References

[1] Steve Myers. The engineering needed for particle physics. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 370(1973):3887–3923, 2012.

[2] Klaus Wille. *The Physics of Particle Accelerators: An Introduction*. Oxford University Press, 2001.

[3] Helmut Vincke, Chris Theis, and Stefan Roesler. Induced radioactivity in and around high-energy particle accelerators. *Radiation Protection Dosimetry*, 146(4):434–439, 2011.

[4] UK Health and Safety at Work etc. Act, 1974.

[5] Claus Grupen. *Introduction to Radiation Protection – Practical Knowledge for Handling Radiaoactive Sources*. Graduate Texts in Physics. Springer-Verlag Berlin Heidelberg, 2010.

[6] F. G. Patterson Jr. *Systems Engineering Life Cycles: Life Cycles for Research, Development, Test, and Evaluation; Acquisition; and Planning and Marketing*, chapter 1, pages 65–115. John Wiley & Sons, 2009.

[7] Karl Ulrich and Steven Eppinger. *Product Design and Development*. McGraw-Hill/Irwin, May 2011.

[8] Karl Wiegers. *Software Requirements*. Microsoft Press, 2 edition, March 2003.

[9] Michael Marszalek, Martin Eder, Andreas Tropschug, Alois Knoll, and Hagen Hoefer. Mobile robots for the simultaneous exploration and 2D determination of radioactivity. In *Proceedings of the 8th WSEAS International Conference on Circuits, Systems, Electronics, Control & Signal Processing (CSECS '09)*. World Scientific and Engineering Academy and Society, WSEAS, December 2009.

[10] R.A. Cortez, X. Papageorgiou, H.G. Tanner, A.V. Klimenko, K.N. Borozdin, and W.C. Priedhorsky. Experimental implementation of robotic sequential nuclear search. In *Mediterranean Conference on Control Automation, MED '07*, pages 1 –6, June 2007.

[11] Ramviyas Parasuraman, Prithvi Pagala, Keith Kershaw, and Manuel Ferre. Energy Management Module for Mobile Robots in Hostile Environments. In Guido Herrmann, Matthew Studley, Martin Pearson, Andrew Conn, Chris Melhuish, Mark Witkowski, Jong-Hwan Kim, and Prahlad Vadakkepat, editors, *Advances in Autonomous Robotics*, volume 7429 of *Lecture Notes in Computer Science*, pages 430–431. Springer Berlin / Heidelberg, 2012.

[12] Craig Larman. *Applying UML and Patterns - An Introduction to Object-Oriented Analysis and Design and Iterative Development*. Pearson Education, Inc., Third edition, 2005.

[13] G. Battistoni, F. Cerutti, A. Fassò, A. Ferrari, S. Muraro, J. Ranft, S. Roesler, and P. R. Sala. The FLUKA code: description and benchmarking. *AIP Conference Proceedings*, 896(1):31–49, 2007.

[14] A. Ferrari, Paola R Sala, A Fassò, and Johannes Ranft. FLUKA: a multi-particle transport code. Technical Report CERN-2005-10, INFN/TC_05/11, SLAC-R-773, CERN, INFN, SLAC, 2005.

[15] Markus Brugger. *The Radiological Situation in the Beam-Cleaning Sections of the CERN Large Hadron Collider (LHC)*. PhD thesis, Technischen Universität Graz, Austria, November 2003.

[16] Joachim Vollaire, Markus Brugger, Doris Forkel-Wirth, Stefan Roesler, and Pavol Vojtyla. Calculation of water activation for the LHC. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 562(2):976 – 980, 2006. Proceedings of the 7th International Conference on Accelerator Applications.

[17] Markus Brugger, Doris Forkel-Wirth, Stefan Roesler, and Joachim Vollaire. Studies of induced radioactivity and residual dose rates around beam absorbers of different materials. In *Proceedings of HB2010*, Morschach, Switzerland, 2010.

[18] G. Dissertori, P. Lecomte, D. Luckey, F. Nessi-Tedaldi, F. Pauss, Th. Otto, S. Roesler, and Ch. Urscheler. A study of high-energy proton induced damage in cerium fluoride in comparison with measurements in lead tungstate calorimeter crystals. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 622(1):41 – 48, 2010.

[19] M. Brugger, H. Khater, S. Mayer, A. Prinz, S. Roesler, L. Ulrici, and H. Vincke. Benchmark studies of induced radioactivity produced in LHC materials, part I: specific activities. *Radiation Protection Dosimetry*, 116(1-4):6–11, December 2005.

[20] M. Brugger, A. Ferrari, S. Roesler, and L. Ulrici. Validation of the FLUKA Monte Carlo code for predicting induced radioactivity at high-energy accelerators. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 562(2):814 – 818, 2006. Proceedings of the 7th International Conference on Accelerator Applications.

[21] Thomas Fabry, Liesbeth Vanherpe, Mathieu Baudin, Chris Theis, Christian Braesch, and Bruno Feral. Interactive intervention planning in particle accelerator environments with ionizing radiation. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 708:32–38, April 2013.

[22] T. Tan, Qi Li, B. Boehm, Ye Yang, Mei He, and R. Moazeni. Productivity trends in incremental and iterative software development. In *Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on*, pages 1 –10, October 2009.

[23] Craig Larman and Victor R. Basili. Iterative and Incremental Development: A Brief History. *Computer*, 36(6):47–56, June 2003.

[24] Python programming language – official website. http://www.python.org.

[25] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit. An Object-Oriented Approach To 3D Graphics*. Kitware, 4 edition, December 2006.

[26] Kitware, Inc. *The VTK User's Guide*, 11th edition, 2010.

[27] Noel Rappin and Robin Dunn. *wxPython in Action*. Manning Publications Co., March 2006.

[28] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns*. Addison-Wesley, 1995.

[29] Thomas Fabry, Christian Braesch, and Bruno Feral. Interactive Visual Intervention Planning – Interactive visualization for intervention planning in particle accelerator environments with ionizing radiation. In *Proceedings of the 8th International Conference on Information Visualization Theory and Applications (IVAPP)*, 2013.

[30] Maaike van den Haak, Menno De Jong, and Peter Jan Schellens. Retrospective vs. concurrent think-aloud protocols: Testing the usability of an online library catalogue. *Behaviour & Information Technology*, 22(5):339–351, 2003.

[31] T. Boren and J. Ramey. Thinking aloud: reconciling theory and practice. *IEEE Transactions on Professional Communication*, 43(3):261–278, 2000.

[32] J.J. Caban, P. Rheingans, and T. Yoo. An Evaluation of Visualization Techniques to Illustrate Statistical Deformation Models. *Computer Graphics Forum*, 30(3):821–830, 2011.

[33] Zhanping Liu, Shangshu Cai, J. Edward Swan II, Robert J. Moorhead II, Joel P. Martin, and T. J. Jankun-Kelly. A 2D Flow Visualization User Study Using Explicit Flow Synthesis and Implicit Task Design. *IEEE Transactions on Visualization and Computer Graphics*, 18(5):783–796, May 2012.

[34] Jibonananda Sanyal, Song Zhang, Gargi Bhattacharya, Phil Amburn, and Robert Moorhead. A User Study to Compare Four Uncertainty Visualization Methods for 1D and 2D Datasets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1209–1218, November 2009.

[35] Florian Lindemann and Timo Ropinski. About the Influence of Illumination Models on Image Comprehension in Direct Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1922 – 1931, December 2011.

[36] Bireswar Laha, Kriti Sensharma, James D. Schiffbauer, and Doug A. Bowman. Effects of Immersion on Visual Analysis of Volume Data. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):597–606, 2012.

[37] M. Levoy. Display of surfaces from volume data. *Computer Graphics and Applications, IEEE*, 8(3):29 –37, May 1988.

[38] Robert A. Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *SIGGRAPH Comput. Graph.*, 22:65–74, June 1988.

[39] Tobias Ritschel. Fast GPU-based Visibility Computation for Natural Illumination of Volume Data Sets. In *Eurographics*, pages 57–60, 2007.