

PERSONAL EXPLORATION ROVER INTEGRATED PLANNING SOFTWARE

The Story of A Program

Aaron Klapheck – California State University

August 2, 2009

Exploration Systems Mission Directorate (ESMD)

Mentor

Alonso Verra, Human Computer Interaction

NASA Ames Research Center

ABSTRACT

This summer program involved creating a computer interface between planning software (SPIFe) and a Personal Exploration Rover (PER). SPIFe is the overarching application and various plug-ins are used in SPIFe to produce the final software products that the customers use. Through developing tests for the SPIFe program Score, experience was gained not only in using the program but also about its development. Tests were developed to determine the full functionality of the program under numerous user conditions and requirements. After learning XML, an activity dictionary was created for the SPIFe program MSLICE, which consists of an XML document detailing the various actions the PER is able to perform. This activity dictionary is then loaded into MSLICE where the activities it contains will be accessible for planning. After developing the activity dictionary the next step taken was to become familiar with the PER programming interface. The PER comes with a fully developed software package that contains all of the commands needed to control the PER's motions and activities (developed in Java). After five weeks of intense study, over 100 Java objects/programs/applets were written (a number of which I will be posting on my website: www.AaronKlapheck.com) and the fundamental aspects of Java were understood. The PER's API was then researched and a number of java methods were created to test the functionality of the PER. After gaining knowledge of the SPIFe programs Score and MSLICE, learning Java, and understanding XML, the SPIFe-PER interface was then undertaken. In anticipation of a plug-in for a SPIFe program, a stand-alone program was created called Personal Exploration Rover Integrated Planning Software (PERIPS). Programming PERIPS involved serialization, event handling, and multithreading of over 20 interrelated objects capable of utilizing the PER's motion, electromagnetic, and image processing capabilities. Once PERIPS is in plug-in format a user would be able to plan activities for the PER in a SPIFe environment and click a button that would have the PER execute the activities right in front of them.

THE PROJECT

Goal

The goal of this program is to demonstrate the capabilities of SPIFe programs in scheduling activities for a Personal Exploration Rover (PER) and to have the PER execute these planned activities.

Exploring the PER

The Personal Exploration Rover (PER) can be succinctly described as a high-tech RC car with the outward appearance of a Mars Rover. The PER has much more functionality than an RC car though, with its image capturing, distance detection, wireless networking, and UV light capabilities it comes at a cost of over \$8,000. The first task undertaken was therefore to become familiar with the PER's capabilities. The manual that came with the PER was downloaded from the PER's website¹ and read along with its associated Java programs and API. After reading the PER documentation a monthly checklist was created to ensure the PER was running at optimum performance throughout the internship and beyond. If the PER was not going to be used for an extended period of time then a storage checklist would be used to make sure it would be stored properly.

After establishing a network connection with the PER, thanks to Alex Eiser, an understanding of the PER's functions was understood. The basic functions of the PER were determined using a calibration program that came with the PER. Another Java program called BasicGUI.java was also used to understand more of the functionality of the PER.

SPIFe Activity Dictionary

SPIFe is a platform used to create user specified planning software applications. How SPIFe programs work basically comes down to taking actions from an activity dictionary (AD), which is a window with a list of activities in it, and placing them in an Excel styled window where starting times, ending times, and various other attributes can be defined. By using this software, all activities of the PER were scheduled. In order for this to occur an AD for the PER was written, a plan was created using these activities, and the plan was exported as an XML document.

An activity dictionary (AD) listing all the PER activities that will be planned, was the first step undertaken. This involved learning XML and utilizing its programming language to create each activity, along with each activity's characteristics and parameters. Mel Ludowise was instrumental in this process as she supplied a number of example ADs from which an understanding of how to structure an AD was acquired. Next a simple plan was created and exported. Before utilization of this plan could be executed by the PER a greater understanding of Java was needed.

Java

Java is used to program all PER commands and to parse the plans created in a SPIFe program. The PER came with an extensive API which detailed all the actions the PER is capable of performing along with the Java syntax used to execute these actions. A large number of example programs also came with the PER to show detailed examples of the Java commands in action. Although a large number of Java programs were given as examples, an understanding of the Java programming language was essential in order to create the number of classes and methods needed to complete the project. All parsing would be done in Java as well, so the task of learning the foundation for the complete Java programming language was undertaken. After four weeks of intensive study and creation of over 100 Java programs, applets, and classes, programming for PER activities was finally undertaken.

A Java object designed to parse the SPIFe plan file was created. Mel Ludowise provided a great deal of help in this area by recommending, and giving examples of, an XPath parser which simplified the task considerable. The XPath parser contained many of the tedious tasks needed to identify the various activities in the plan XML document. Once identification of the activities contained in the plan was accomplished, the next step was running the activities on the PER.

A Java object was created that took the identified activities and transformed them into the format understandable by the PER. This object also establishes a network connection which would be used to transmit these transformed activities to the PER for execution. This program involved serialization, polymorphism, and multithreading of over 20 interrelated objects capable of utilizing the PER's motion, electromagnetic, and image processing capabilities. Finally another object was created to generate a log of the PER's activities so a user could keep track of the commands the PER actually executed.

Program Creation

The Java objects created that would allow the planned activities to be executed by the PER, were designed, created, and tested, but were not in a form easily accessible or understandable by the user. The next step was to create an environment that would encapsulate the functionality of both objects in a form easy for a user to understand. The GUI environment created was Personal Exploration Rover Integrated Planning Software (PERIPS). This program was designed, hand-coded, and tested to ensure optimal performance. User testing was then performed on the program and numerous revisions of the program were created. The revision process occurred over the course of two weeks until the final PERIPS program as it looks now was released.

CONCLUSION

Through experience gained in learning Java, creating plans in SPIFe, and controlling the PER, the interface program PERIPS was created. This program demonstrates an application of a plan through its execution by the PER. PERIPS works well as a standalone application, but there are a number of things to be improved upon. Although functional as a standalone program it would be preferable to have PERIPS as a plug-in for a SPIFe program. This way after a user is done planning their activities for the PER, they could simply hit a button while still in SPIFe and watch the PER execute the user specified activities. The current design of PERIPS is solely with function in mind, for later revisions form could also be considered.