IN-61
432 253

IAF–98–U.3.04
Telescience Resource Kit Software Lifecycle

C. Griner and M. Schneider
NASA Marshall Space Flight Center
MSFC, AL, USA

49th International Astronautical Congress
Sept. 28–Oct. 2, 1998/Melbourne, Australia

TELESCIENCE RESOURCE KIT SOFTWARE LIFECYCLE

C. Griner and M. Schneider
NASA Marshall Space Flight Center
Marshall Space Flight Center, Alabama

Abstract

The challenge of a global operations capability led to the Telescience Resource Kit (TReK) project, an in-house software development project of the Mission Operations Laboratory (MOL) at NASA's Marshall Space Flight Center (MSFC). The TReK system is being developed as an inexpensive comprehensive personal computer- (PC-) based ground support system that can be used by payload users from their home sites to interact with their payloads on board the *International Space Station* (*ISS*). The TReK project is currently using a combination of the spiral lifecycle model and the incremental lifecycle model. As with any software development project, there are four activities that can be very time consuming: Software design and development, project documentation, testing, and umbrella activities, such as quality assurance and configuration management. In order to produce a quality product, it is critical that each of these activities receive the appropriate amount of attention. For TReK, the challenge was to lay out a lifecycle and project plan that provides full support for these activities, is flexible, provides a way to deal with changing risks, can accommodate unknowns, and can respond to changes in the environment quickly. This paper will provide an overview of the TReK lifecycle, a description of the project's environment, and a general overview of project activities.

1. BACKGROUND

In the past, most engineers and scientists who participated in payload operations ground support activities were required to travel to a major NASA mission operations control center to carry out their duties. As we enter the *ISS* era, this is no longer feasible. In the 1980's and 1990's a typical Spacelab mission lasted a few days. In the future, payload activities in support of a payload increment aboard the *ISS* are expected to last a few months and some payloads will be on board the Space Station for many years. The financial costs associated with requiring personnel to travel to a major control center to provide support for these activities is infeasible. For these reasons, NASA is now implementing a new ground operations philosophy called Remote Operations. Remote Operations provides a way for engineers and scientists to conduct payload operations from their home sites.

The Payload Operations Integration Center (POIC), located at MSFC in Huntsville, Alabama, is the main control center for *ISS* payload operations. However, most of the engineers and scientists participating in payload operations activities will not be located in Huntsville but at home sites all over the world. In order to monitor and control payloads aboard the *ISS* from anywhere in the world, NASA needed an inexpensive portable ground control system that could easily be installed at sites around the world. The system needed to be simple, secure, and inexpensive to design, develop, distribute, and maintain. MSFC's TReK is meeting that need.

2. TReK PRODUCT DESCRIPTION

A TReK system consists of a commercial PC configured with commercial off-the-shelf (COTS) software, shareware, freeware, and MSFC MOL-provided software to provide a complete ground control system for *ISS* payload users. A TReK system is not a stand-alone ground control system but one that interfaces with a major control center, such as the POIC, in order to provide a complete set of ground control system capabilities. Capabilities provided by TReK include the capability to view telemetry, perform local exception monitoring, local calculations, word processing, file management, and local command and control. TReK users can extend TReK capabilities by using the TReK application programming interface with COTS products to utilize local telemetry and command functions. TReK systems also provide multiple levels of ground system checkout, including a stand-alone training mode, the ability to interface with a payload through a Suitcase Simulator system while the payload is still in a ground-based laboratory, and the ability to conduct interface tests with a supporting facility, such as the POIC. TReK users have complete control over their system and are responsible for system configuration, system management, and security. A complete TReK system costs less than $10,000.

3. TReK PROJECT OVERVIEW

TReK, an MSFC MOL in-house software development project, has been ongoing since 1996 and is scheduled to be completed in 2002. In order to understand the TReK software lifecycle and why it was selected, it is important to understand some of the characteristics of the TReK project. Many factors affected the selection of the TReK software lifecycle that is currently in place. Some of the primary factors include:

- Personnel resources
- Budget resources
- Ground system development knowledge
- Software development knowledge/skills
- Software development environment
- Interfaces with other large complex systems
- Schedule constraints.

The TReK development team was formed in 1996 and is responsible for all aspects of the TReK project, including project management, procurements, requirements, design, development, testing, documentation, distribution, and marketing. Since there are only eight team members, each team member must be able to perform a wide variety of tasks.

The members of the TReK development team are part of the organization responsible for the design and development of NASA's POIC. Most of the requirements for the TReK system were derived from the POIC requirements. Almost all TReK team members worked for 5 or more years on the design and development of the POIC before beginning work on the TReK project. Therefore, the TReK team began the project with a substantial knowledge of ground control systems and a unified vision of what the TReK product needed to be.

Although the team had substantial knowledge about ground control systems, they were faced with a new development environment. Most had worked in the POIC development environment and were well versed in UNIX, C, X-Windows, Motif, and Oracle. However, to develop the TReK system required that they become knowledgeable about a new development environment which included Windows NT, C++, and Microsoft Access. This introduced a development environment learning curve at the beginning of the project.

There were also schedule constraints that were in place at the beginning of the project. Since the TReK system had to interface with other ground control systems, and be ready to support the *ISS* payload schedule, the TReK schedule was somewhat dictated by the other larger system's schedules that were already in place.

In addition to the factors above, the normal software development project activities had to be considered. Four activities that can be very time consuming are software design and development, project documentation, testing, and umbrella activities, such as quality assurance and configuration management. In order to produce a quality product, it is critical that each of these activities receives the appropriate amount of attention. For TReK, the challenge was to lay out a lifecycle and project plan that addressed all the factors listed above, provided full support for normal software development activities, was flexible, provided a way to deal with changing risks, could accommodate unknowns, and could respond to changes in the environment quickly.

4. TReK PROJECT LIFECYCLE

The TReK software lifecycle, shown in Fig. 1, is a combination of the incremental development model and the spiral model. To understand why this lifecycle was selected, one must understand the fundamental concepts behind both types of lifecycles and how they relate to the TReK project factors described in section 3.

Pressman[1] describes the pure incremental model as follows: "The incremental process model, like prototyping and other evolutionary approaches, is iterative in nature. But unlike prototyping, the incremental model focuses on the delivery of an operational product with each increment. Early increments are "stripped down" versions of the final product, but they do provide capability that serves the user and also provide a platform for evaluation by the user."

The incremental model was well suited for TReK for three main reasons: External interfaces, early product availability for customers, and personnel resources. As stated earlier, the TReK system interfaces with the POIC system. The POIC development schedule was split into five separate builds, and many of the interfaces needed by the TReK system were being delivered in different builds. Due to the *ISS* schedule, it was not feasible to wait until the POIC was completely delivered before beginning work on the TReK product. It was important to provide users with access to the TReK product as soon as possible; therefore, the TReK lifecycle naturally fit into an incremental pattern. Additionally, one of the main factors that affected the TReK project was personnel and budget resources. By developing TReK in an incremental fashion,

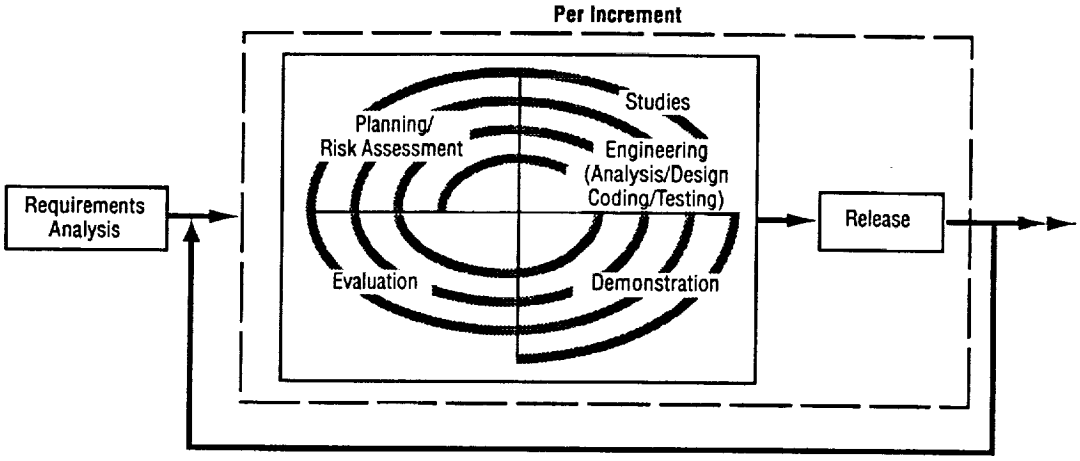

Fig. 1. TReK Software Lifecycle.

it was possible to define a project plan that aligned with personnel and budget resources that had been allocated. For these reasons, TReK incorporated the incremental lifecycle model into the TReK lifecycle.

Pressman[2] describes the spiral model as follows: "The spiral model is divided into a number of framework activities, also called task regions. Typically, there are between three and six task regions. Each of the regions is populated by a series of work tasks that are adapted to the characteristics of the project to be undertaken. For small projects, the number of work tasks and their formality is low. For larger, more critical projects, each task region contains more work tasks that are defined to achieve a higher level of formality. In all cases, the umbrella activities (e.g., software configuration management and software quality assurance) are applied. As this evolutionary process begins, the software engineering team moves around the spiral in a clockwise direction, beginning at the core. The first circuit around the spiral might result in the development of a product specification; subsequent passes around the spiral might be used to develop a prototype and then progressively more sophisticated versions of the software. Each pass through the planning region results in adjustments to the project plan. Cost and schedule are adjusted based on feedback derived from the customer evaluation. In addition, the project manager adjusts the planned number of iterations required to complete the software. The spiral model is a realistic approach to the development of large-scale systems and software. Because software evolves as the process progresses, the developer and customer better understand and react to risks at each evolutionary level. The spiral model uses prototyping as a risk reduction mechanism, but more important, enables the developer to apply the prototyping approach at any stage in the evolution of the product. It maintains the systematic stepwise approach suggested by the classic life cycle, but incorporates it into an iterative framework that more realistically reflects the real world. The spiral model demands a direct consideration of technical risks at all stages of the project, and if properly applied, should reduce risks before they become problematic."

As with the incremental model, there are multiple reasons the spiral model was well suited to TReK. It was important for the TReK project to address all the key issues associated with a software development project, including documentation, configuration management, testing, and other quality assurance activities. However, the TReK project faced many issues, including a new development environment, limited personnel resources, a tight schedule, and dependencies on external systems, such as the POIC. All of these issues translated into risks that, if handled incorrectly, could have destroyed the project. The TReK lifecycle had to provide a way to deal with existing and future risks that would evolve as the product evolved. It also had to provide flexibility in dealing with current unknowns, such as the new development environment, and the ability to respond quickly to changes when they occurred, such as changes in external interfaces. Since TReK had to interface with several external systems, any changes in their schedules or technical direction would have an impact on the TReK schedule and direction. The TReK project team had to be in a position at all times to change their plans and schedules accordingly. Since the spiral model provided a way to address all of these issues, it was incorporated into the TReK lifecycle.

As shown in Fig. 1, the TReK lifecycle has three main components: Requirements analysis, the spiral (containing four task regions), and release. The spiral and release

portions of the lifecycle are applied per increment. There are currently four increments planned. The number of spirals executed within each increment is not fixed but determined during the course of each increment and is dependent on the state of the product and the current set of risks that must be addressed. This is the primary reason the spiral lifecycle provides flexibility. This arrangement makes it possible to address unknowns and respond quickly to changes in the environment.

4.1 Requirements Analysis

The requirements analysis block represents the initial software requirements review conducted for the TReK project. This activity established the overall project requirements and the conceptual framework for the project. The development environment and final product environment were also identified during this timeframe. As stated earlier, many of these requirements were derived from existing POIC requirements. The requirements analysis block is shown as an entry point to the main part of the lifecycle to show what actually occurred for the TReK project. This does not imply that requirements will not change in the future. When requirements change, these changes are handled by activities within the spiral portion of the lifecycle.

4.2 Spiral

Although they are not visible, the spiral model contains all the traditional "waterfall" activities, including analysis, design, coding, testing, and maintenance. It also includes all other project activities, including planning, scheduling, system management, and umbrella activities such as configuration management and quality assurance. These activities are all addressed within the four main task regions in the TReK spiral. The purpose of each task region is described below.

Planning/Risk Assessment The planning/risk assessment quadrant is used to plan the high-level activities that will be addressed during a spiral pass. These activities are recorded in the TReK Spiral Plans & Results document. Two major types of activities are addressed: Activities needed to evolve the product to the next level and activities needed to address current risks. Each spiral plan identifies current risks, prioritizes the risks, and defines high-level activities that can be performed to reduce each risk.

Studies and Engineering The studies and engineering quadrant is where the majority of work takes place. This quadrant is used to perform studies and other activities that address current project tasks and risks, and is also used to perform the engineering necessary to move the product to the next level of maturity. Many different types of engineering activities are addressed in this quadrant, including requirements analysis, design, coding, and testing. The team enters this quadrant with the high-level list of activities that were defined in the planning/risk assessment quadrant and documented in the spiral plan. The details associated with each activity are reflected in the TReK Current Activities Schedule. The TReK Current Activities Schedule is used internally by the TReK team members to track day-to-day activities and addresses the following areas:

- Project documentation
- Technical reviews
- Configuration management
- External interfaces
- Design/development activities
- Interface definitions and deliverables
- System issues
- System management
- Meetings
- Internal and external tests
- Support activities
- Studies and evaluations
- Training
- Team actions
- Procurements
- Software release activities.

The Current Activities Schedule identifies deadlines for all activities, including design, development, and integration activities as deemed appropriate to meet the product goals of the spiral that is in progress. As activities are completed, they are recorded in the TReK Completed Tasks List. Additionally, a TReK To Do List is maintained that tracks activities that need to be addressed in the future. The TReK To Do List is fed back into the planning/risk assessment quadrant at the beginning of the next spiral. A TReK Dependencies List is also maintained that identifies dependencies on external entities, such as the POIC system and the Suitcase Simulator system, that could affect the TReK schedule.

Demonstration The demonstration quadrant is used by the TReK team to demonstrate what was accomplished during the spiral. This may include an internal spiral briefing to management as well as a product demonstration. If applicable, a beta copy of the product is released to the TReK user community at this time. This only occurs if the beta release is a planned event that is part of the current spiral plan.

Evaluation The evaluation quadrant is used to evaluate the project's progress. Conclusions reached during this quadrant will be fed back into the next spiral's planning/ risk assessment activities. In general, evaluation activities are limited to internal assessments. However, in some cases, it does include inputs from TReK users if a beta copy of the product was released during the spiral.

4.3 Release

The release block represents the formal delivery of an increment to the user community, including all documentation and software associated with the increment. Although maintenance is not explicitly shown, it is an integral part of the lifecycle and is handled within the spiral. As changes are needed, they are planned and executed using the four quadrants of the spiral.

5. PROJECT DOCUMENTATION

Software projects are comprised of much more than just software. Project documentation is an extremely important component of any software product, since it often serves as one of the primary tools used to train new project personnel and is used to assist maintenance personnel. TReK documentation consists of both internal and external documents. Some of the internal documents include the TReK Project Plan, TReK Project Standards, TReK Testing Plan, the suite of TReK design documents, the TReK Incremental Development Plan, and TReK schedules.

Any document which is distributed to the user community is considered an external document. These include the TReK Requirements document, the suite of TReK user guides and tutorials, and any user reference manuals. All TReK project documents are identified in the TReK Project Documents, a living document updated as new project documents are identified. The Project Documents contains the following information for each document: A description, document number, configuration management policy, the point of contact, a list of internal reviewers, signature authority, if applicable, and a final distribution list. All the documents identified in the TReK Project Documents are considered an integral part of the overall TReK Project Plan.

6. UMBRELLA ACTIVITIES

The fundamental TReK project goal is to produce a quality product. There are multiple umbrella activities that must be performed to meet this goal. To address this, the TReK lifecycle includes the following activities: Technical reviews, standards, a test plan, mechanisms for record keeping and reporting, and configuration management activities. The TReK team holds internal reviews during each spiral to review the design, code, and documentation associated with the product that is being developed during that spiral. All aspects of the TReK product are subject to standards. The TReK Project Standards document defines the documentation standards, user interface design standards, and coding standards. The TReK Test Plan defines the procedures associated with executing unit testing, integration testing, and validation testing. Records are kept via the Completed Tasks List discussed in section 4.2 and through a variety of TReK documentation. Reporting will be discussed in section 7. All TReK products, including documentation and software, are under configuration management using a commercial configuration management tool. TReK software is controlled on a spiral basis. After each spiral, a new branch is created and the code is moved to begin work on the next spiral. Each of the umbrella activities discussed above is important to the overall quality of the TReK product, and is a fundamental part of the TReK lifecycle.

7. PROJECT REPORTING

Project reporting is an important part of any software project. TReK project reporting occurs at many different levels, so that all parties involved with the project stay informed. Each of the five different reporting mechanisms used is described below.

Written Weekly Report A written report documenting any important events or progress made during the week is submitted to internal management personnel and NASA's *ISS* Program office personnel on a weekly basis.

Written Twice-Monthly Milestones Report A milestones report, submitted to internal management twice monthly, reflects the current status of design and development, documentation, and testing.

Written Monthly Project Metrics Report At the end of each month, a written status report is submitted to internal management personnel and NASA's *ISS* Program office personnel, documenting significant events or issues that occurred during the month. This report also includes an accumulated milestones chart for the current spiral and the TReK Project Schedule. The TReK Project Schedule tracks progress on an increment level.

Monthly Status Briefing Each month a status briefing is provided in a forum that includes internal management,

NASA's *ISS* Program office personnel, and the user community. This briefing usually covers activities that are of interest to the user community.

Spiral Briefings Spiral briefings, internal meetings held with MOL management at the end of each spiral, include both a briefing and demonstration of the evolution of the product that occurred during the spiral.

8. CUSTOMER (USER) COMMUNICATION

In comparison to other software lifecycles often used on government software projects, the TReK project is being handled much more like a commercial venture. TReK customers have access to the initial requirements document, but instead of attending open preliminary design reviews (PDR's) and critical design reviews (CDR's), they instead learn and communicate about the TReK product through beta software releases. During early project planning, it was clear that PDR's and CDR's would put a significant strain on schedule and personnel resources. Additionally, these types of reviews are often filled with detailed internal design information that is often irrelevant to the customer. In comparison, the beta software program provides the customer with early access to the product and a communication method which is helpful to both the customer and the development team.

There are multiple ways in which the customer can communicate about the TReK product. The TReK product is advertised through normal *ISS* program channels as well as through the TReK web site (http://snail.msfc.nasa.gov/trek/trekfrme.htm). The TReK web site contains a variety of information about the TReK product.

Potential TReK customers can provide inputs about the TReK product through multiple forums. If there is a general requirement missing, they can write an Engineering Change Request against the TReK Requirements document. For firsthand interaction with the TReK product, they can participate in the TReK Beta Software Testing program. Each TReK beta release consists of a complete set of documentation, software, and example programs. This allows each customer to interact with the product while it is still being developed and offer inputs during this time. Beta participants can submit comments, questions, or problem reports through the TReK Customer Input form, submitted through the TReK Help Desk. All Customer Input forms submitted are tracked internally, and each customer receives a response to their input. These inputs are used by the TReK team to help guide the vision of the TReK product and develop better user documentation.

9. PRODUCT DISTRIBUTION

The TReK web site is a fundamental part of the TReK product and the primary means by which information is distributed and production distribution made to potential and existing customers. Since TReK is a heavily COTS-based solution, TReK users are expected to purchase their own PC and MOL-recommended COTS products. Once they have configured their PC, they can then go to the TReK web site to download the software, simply by doubleclicking on the self-extracting file. The TReK installation program installs all TReK software, documentation, and example programs. This saves the *ISS* program a great deal of money, since there are no shipping costs involved. It also makes the product readily available to the customer at any time.

10. SUMMARY

The TReK project is currently in its third spiral. The first increment is expected to be completed in the summer of 1999. Although the project is only 2 years old, the software lifecycle selected has proven very beneficial and is standing the test of time. So far, the TReK project has stayed on schedule and has already completed its first beta release to the *ISS* payload community. With only eight people and such a large set of project activities, it is very easy to see how a project team would be inclined to begin dropping important umbrella activities. However, the TReK lifecycle is very flexible and makes it possible to move tasks around so they can be addressed at an appropriate time, based on the current set of project activities and risks. Many other lifecycles have fixed timeframes for each activity. This often locks a software team into a timeframe that does not align with other important activities that need to be addressed, and often leads to schedule slips and poor team morale. The TReK lifecycle provides a way to avoid these types of problems by providing solid structure with the flexibility needed to meet changing priorities and risks.

REFERENCES

1. Pressman, R.S., "Software Engineering—A Practitioner's Approach," McGraw-Hill, p. 37, 1997.

2. Pressman, R.S., "Software Engineering—A Practitioner's Approach," McGraw-Hill, pp. 39–40, 1997.