A PROPOSAL ON USING REUSE READINESS LEVELS TO MEASURE SOFTWARE REUSABILITY

Robert R. Downs [1*] ***and James J. Marshall*** [2]

[*1]*Columbia University, Center for International Earth Science Information Network (CIESIN), 202 Geoscience, Lamont Campus, Palisades, NY, 10964, USA*
Email: rdowns@ciesin.columbia.edu
[2]*INNOVIM / NASA Goddard Space Flight Center, Mail Stop 614.9, Greenbelt, MD, 20771, USA*
Email: James.J.Marshall@nasa.gov

ABSTRACT

The use of scientific data is becoming increasingly dependent on the software that fosters such use. As the ability to reuse software contributes to capabilities for reusing software-dependent data, instruments for measuring software reusability contribute to the reuse of software and related data. The development and current state of a proposed set of Reuse Readiness Levels (RRLs) are summarized, and potential uses of the software reusability measures are described, along with proposed use cases to support sponsorship of software projects, software production, software adoption, and data stewardship during the systems development lifecycle and the data lifecycle.

Keywords: Software reuse, Reuse readiness, Reuse metrics, Use cases, Scientific data, Data reuse

1 INTRODUCTION

Throughout the data lifecycle, scientific data and research-related information often require the development or adoption of systems and software to facilitate data creation and use. In many cases, data are generated, found, accessed, visualized, and analyzed using capabilities provided by software (National Science Foundation Cyberinfrastructure Council, 2007). Furthermore, as scientific practices evolve, communities engaged in e-science often need system tools and software services for discovering, analyzing, and integrating data or to create new data products and services for using data (McGuinness, Fox, Cinquini, West, Garcia, Benedict, et al., 2008). Systems and software components also are needed for managing and preserving data to enable their use by current and future communities of users (Berman, 2008). Such requirements demonstrate how the use and reuse of scientific data are becoming increasingly dependent on the systems and software that foster these uses. In many cases, the use of data might not be feasible without the capabilities provided by the software on which the data are dependent.

The challenges facing the reuse of scientific data and research-related information are being recognized, and capabilities for managing and preserving scientific data are being improved to enable e-science practices (Anderson, 2004; Consultative Committee for Space Data Systems, 2002; Jacobs & Humphrey, 2004; National Science Board, 2005; To Stand the Test of Time, 2006). Such improvements contribute to data stewardship practices throughout the data lifecycle across various disciplines and include, among other contributions, developments in policies, informatics, and cyberinfrastructure that foster interoperability for scientific data reuse in terms of data formats (Albani & Giaretta, 2009; Peterson, 2009; Woods & Brown, 2008; Yamagishi, Nagao, Suzuki, Tamura, Hatakeyama, Yanaka, et al., 2009), semantics (Fox, McGuinness, Cinquini, West, Garcia, Benedict, et al., 2009; Hu, Ouyang, Wu, Zhang, & Zhao, 2009; Knight & Pennock, 2009; Ojala & Over, 2009; Thieman, Roberts, King, Harvey, Perry, & Richards, 2010; Tzitzikas, 2009; Zhang, Hu, & Li, 2009), and operational practices (Berman, 2008; Downs & Chen, 2010; Kühne & Liehr, 2009; Rajasekar, Moore, Wan, & Schroeder, 2010; Uhlir, Chen, Gabrynowicz, & Janssen 2009).

While progress on enabling data reuse, fostering interoperability, and facilitating scientific data stewardship is promising for e-science, challenges for the preservation of scientific data and research-related information must continue to be addressed (Barton, Smith, & Weaver, 2010; Dubin, Futrelle, Plutchak, & Eke, 2009; Interagency Working Group on Digital Data, 2009; Janée, Frew, & Moore, 2009; National Science Foundation Cyberinfrastructure Council, 2007). Such challenges include the dependencies of scientific data upon the software and information systems used to create and provide access to the data (Matthews, Shaon, Bicarregui, Jones, Woodcock, & Conway, 2009). Complementing the progress that has been achieved for data reuse,

methods for enabling the potential reusability of software and other system artifacts also can offer support by addressing the dependency of data upon software and system artifacts.

In consideration of the dependencies of data upon systems and software, the management of scientific data and technical information often requires managing aspects of the systems and software used to create or work with the data so that these artifacts can be reused. Similarly, it may be necessary to preserve the capabilities provided by the systems and software, upon which the use and reuse of data are dependent, so that such capabilities can be reused to enable future use of the data. In recognition of these considerations, facilitating the reuse of software artifacts and the capabilities offered by such artifacts becomes an integral aspect of scientific data stewardship when use of the data is dependent on systems or software artifacts.

The risks associated with the software dependencies of scientific data should be considered and managed throughout the entire data lifecycle. Those who create data or integrate existing data to generate new data products should consider the effects of such dependencies when choosing to use systems or software for a data project. Likewise, organizations that accept the responsibility for managing and preserving scientific data should identify any dependencies associated with the data that they manage and identify approaches for managing such dependencies for the data within their care. Documenting data dependencies is a fundamental aspect of data provenance (Cheney, Ahmed, & Acar, 2007; Moreau, Clifford, Freire, Gil, Groth, Futrelle, et al., 2009; PREMIS Editorial Committee, 2008). Identifying and managing software artifacts, which present dependencies for scientific data, offer an approach to data stewardship to complement existing data reusability measures.

In some cases, it becomes necessary to reuse some software in order to use data that are dependent on such software for their use. For example, describing the software and version previously used can facilitate replication or evaluation of data from previous studies (Agosti, Di Nunzio, & Ferro, 2007). It also may be necessary to store the same version of the needed software with the data (Lynch, 2008). Recommendations for the management of models and modeling data also advocate storing the software with the data (Hill, Crosier, Smith, & Goodchild, 2001; Cavalcanti, Mattoso, Campos, Llirbat, & Simon, 2002; Hunter, 2006). Such solutions can help to mitigate the risks to data that are associated with software dependencies if the software can be reused and if any other dependencies that are required by the software also are addressed.

Facilitating the reusability of software can help to ensure that needed software can be reused as the technology and uses for the data evolve. Changes in hardware, operating systems, and software applications can present challenges for the reuse of data that are dependent on legacy software either for rendering the data or for understanding how the data were generated. Similarly, changes in the way that scientific data are used or integrated can impose additional requirements for the software that facilitate access to the data. Considering such challenges, the reusability of the systems and other software artifacts, upon which use of the data is dependent, becomes increasingly important for effectively and efficiently managing the data to enable their use in the future. When managing and preserving these data, it will be necessary to manage and preserve the software capabilities upon which the data are dependent. In such cases, data stewardship also encompasses stewardship for the software as well as for other artifacts that are needed to use the data. Tools for managing the reusability of software can complement existing data stewardship capabilities of organizations and individuals responsible for managing the data or other digital assets that are dependent on such software.

Measures for determining the potential reusability of software can facilitate the sponsorship, production, adoption, and stewardship of software that is being considered for potential reuse. Recognizing the criteria for reusability can offer capabilities for evaluating and measuring the potential reusability of software investments and provide guidance for evaluating and improving the potential reusability of systems, software, and related components. Assessing the extent to which software meets the criteria for potential reusability could inform decisions regarding the sponsorship, production, adoption, and stewardship of software and the data that rely on such software.

Measures for software reusability are proposed to improve capabilities for software project sponsors, software developers, software adopters, and data stewards to evaluate the potential reusability of systems, software components, and related artifacts. Existing measurements of technology maturity are described in Section 2 with justification for a new measure focused on reusability. Development of the Reuse Readiness Levels (RRLs) is described in Section 3, and draft RRLs summary descriptions are presented. Potential uses of the RRLs and their topic area levels are introduced in Section 4. Use cases for using the RRLs throughout the systems development lifecycle and the data lifecycle are proposed in Section 5. Progress to date is summarized in Section 6, and conclusions are offered in Section 7.

2 TECHNOLOGY MATURITY MEASUREMENTS

Software reuse can offer benefits for organizations engaged in software development and for downstream users when the reuse of software components is systematic (Mohagheghi & Conradi, 2007). Tools for measuring the potential of software components for reuse could contribute to the efforts of both producers and adopters of reusable software components by providing capabilities for determining the extent to which software components of interest meet the conditions for reuse. Similarly, they could assist organizations that sponsor the production and/or adoption of reusable software components by informing the requirements of the desired product, and they can assist data stewards in their efforts to ensure that any software dependencies the data have are appropriately addressed in the data archives. For example, Technology Readiness Levels (TRLs) are a commonly used tool for measuring technological maturity although there are other similar measurements as well. These measures are inclined to focus on the maturity of the technology as a whole and thus may not be the best tool to use for assessing the reusability of individual software components for potential reuse. Recognizing this gap, the National Aeronautics and Space Administration (NASA) Earth Science Data Systems (ESDS) Software Reuse Working Group (WG) has proposed Reuse Readiness Levels (RRLs) as a new tool to help alleviate potential problems faced in the absence of an instrument for measuring software reusability maturity.

2.1 Background on Existing Measurements

TRLs were originally described by Sadin, Povinelli, and Rosen (1989), but the nine-level form in which they are currently used was depicted by Mankins (1995) as "a systematic metric/measurement system that supports assessments of the maturity of a particular technology and the consistent comparison of maturity between different types of technology". The levels of the TRLs range from basic research in under-developed technologies to developed technologies, such as those used in system launch and operations. Although originally designed with a focus on hardware, the TRLs have successfully been applied to software as well (NASA, 2008a). In addition, NASA has recently completed an effort to update the TRLs to specifically include separate descriptions for applicability to hardware and software as well as to provide the criteria that must be met for a technology to mature up the scale from one TRL to the next (NASA, 2008b). The NASA definitions of TRLs are used frequently, along with those of the United States Department of Defense (DoD), detailed in the Technology Readiness Assessment (TRA) Deskbook (DoD, 2009). However, TRLs are not the only method or tool for assessing technology maturity, and many other scales have been designed by other organizations for the same general purpose as TRLs (Graettinger, Garcia, Siviy, Schenk, & Van Syckle, 2002; Smith, 2004a, 2004b, 2005; Gold & Jakubek, 2005; OPEN Process Framework, 2005; OpenBRR.org, 2005).

2.2 Justification for a New Reusability Measurement

Considering the potential value of reusing previously developed components, in terms of attaining efficiencies and improving the quality of systems being developed, the availability of tools to support the selection of software components for potential reuse would be beneficial. Furthermore, the availability of such tools could improve capabilities for producing reusable software. Thus measures for determining the reusability of software components would be beneficial for both producers and adopters of reusable software and also could be used by the organizations sponsoring such production and adoption activities. They may be used, furthermore, by data stewards who must ensure that software dependencies are addressed in their data archives. The previously mentioned technology and software maturity measures, however, have a tendency to focus on the overall maturity of software or technology and do not place much emphasis on the characteristics of the technology that could facilitate reuse. If it is considered at all with these measures, the reusability of software artifacts is typically only measured in a limited way. For instance, the TRL descriptions from the OPEN Process Framework's TRLs (2005) factor reuse into their levels only for reused critical technologies; reuse is not a part of the levels for non-critical technologies. Some metrics do focus on aspects of reuse, and they have been assessed in terms of their potential for determining the reusability of software components (Sharma, Grover, & Kumar, 2009).

The results of surveying Earth science community members about their experiences and practices in reuse revealed that technology reuse is performed primarily to attain savings in costs or time and to enable product reliability (Marshall, Olding, Wolfe, & Delnore, 2006). Software adopters would be aided in their efforts to achieve these goals by having an instrument to evaluate and quantify the readiness of software assets for reuse. The process of finding, assessing, and selecting software assets that are reusable also could become more efficient as a result of the availability of instruments to measure reusability (Marshall & Downs, 2008). Therefore, Reuse Readiness Levels (RRLs) have been proposed to address these issues.

3 DEVELOPMENT OF THE REUSE READINESS LEVELS

3.1 Identification of Topic Areas

While developing RRLs, the WG used the structure of NASA's TRLs (Mankins, 1995) as a guide because the TRLs have been applied successfully and have proven to be a useful measure for their domain. The levels of the nine topic areas identified as measures to evaluate reusability include "Documentation, Extensibility, Intellectual Property Issues, Modularity, Packaging, Portability, Standards Compliance, Support, and Verification and Testing" (Marshall, Downs, Samadi, Gerard, & Wolfe, 2008, p. 304; NASA ESDS Software Reuse WG, 2008b). In alignment with the TRLs, the topic area levels were constructed with nine rating categories for measuring the potential reusability of a resource in each of the nine topic areas. The overall RRLs were created by combining and summarizing these levels across all topic areas.

3.2 Development of Topic Area Levels and RRLs

The development of the measures and the scales of the RRLs has been completed in an iterative manner, consisting of peer-review cycles conducted by the WG in cooperation with the community of Earth science data system developers (Marshall & Downs, 2008). Initial reviews of the early draft topic levels were conducted during the 2007 ESDS WG Meeting, where the RRLs and the topic area levels were presented (NASA ESDS Software Reuse WG, 2007a) and displayed during a poster session (NASA ESDS Software Reuse WG, 2007b). Based on the feedback and recommendations received, a set of comprehensive descriptions was composed to describe the brief labels that had been developed previously.

Presentations of this updated work by Marshall, Berrick, Bertolli, Burrows, Delnore, Downs, et al. (2007) and the NASA ESDS Software Reuse WG (2008a) resulted in additional recommendations for improving the topic area levels followed by independent reviews for each topic area. Subsequent reviews during the 2008 ESDS Working Group Meeting produced more revisions to the RRLs and to the topic area levels (NASA ESDS Software Reuse WG, 2008b). Consequently, the RRLs Version 1.0 document was completed during a workshop in 2010 and has been released (NASA ESDS Software Reuse Working Group, 2010). The summary descriptions of the RRLs are presented in Table 1.

After being released, the RRLs continue to be reviewed to identify areas for potential improvement. The continuing review of the RRLs is focusing on improvements of their precision for reliably measuring the state of a particular software component that is being assessed. Similarly, the RRLs are being reviewed to ensure that each summary level can be operationally distinguished from nearby summary levels in an objective manner so that the distinction between each level is clear and can be determined readily by a software producer or by a potential user of candidate software. One way to enable clear distinctions for categorizing software components would be to establish a few general categories, consisting of multiple levels, which then could be used to indicate the general state of a software component that is being evaluated for potential reuse. Possible categories could reflect states of potential reusability that a software component has achieved, such as reuse not enabled, reuse partially enabled, and reuse enabled.

In addition to identifying improvements for their precision and usefulness as measures of reusability, the review of the RRLs V1.0 document is needed to ensure that the summary descriptions for each level accurately reflect the criteria for each of the nine topic areas. The RRLs also are being reviewed to determine whether consideration for algorithms and functions are complete, whether the design is adequate, whether migration between run-time environments are addressed, whether recent advances in underlying technologies are considered, whether the degree of real-time needs of the target system are considered, and whether alternative support capabilities are considered. Improvements that are identified for the RRLs will be included in future versions.

Table 1. Reuse Readiness Levels Version 1.0 (adapted from Marshall, Downs, & Samadi, 2010, p. 30)

RRL	Summary	Description
1	Limited reusability; the software is not recommended for reuse.	Little is provided beyond limited source code or pre-compiled, executable binaries. There is no support, contact information for developers or rights for reuse specified, the software is not extensible, and there is inadequate or no documentation.
2	Initial reusability; software reuse is not practical.	Some source code, documentation, and contact information are provided, but these are still very limited. Initial testing has been done, but reuse rights are still unclear. Reuse would be challenging and cost-prohibitive.
3	Basic reusability; the software might be reusable by skilled users at substantial effort, cost, and risk.	Software has some modularity and standards compliance, some support is provided by developers, and detailed installation instructions are available, but rights are unspecified. An expert may be able to reuse the software, but general users would not.
4	Reuse is possible; the software might be reused by most users with some effort, cost, and risk.	Software and documentation are complete and understandable. Software has been demonstrated in a lab on one or more specific platforms, infrequent patches are available, and intellectual property issues would need to be negotiated. Reuse is possible, but may be difficult.
5	Reuse is practical; the software could be reused by most users with reasonable cost and risk.	Software is moderately portable, modular, extendable, and configurable, has low-fidelity standards compliance and a user manual, and has been tested in a lab. A user community exists but may be a small community of experts. Developers may be contacted to request limited rights for reuse.
6	Software is reusable; the software can be reused by most users although there may be some cost and risk.	Software has been designed for extensibility, modularity, and portability, but software and documentation may still have limited applicability. Tutorials are available, and the software has been demonstrated in a relevant context. Developers may be contacted to obtain formal statements on restricted rights or to negotiate additional rights.
7	Software is highly reusable; the software can be reused by most users with minimum cost and risk.	Software is highly portable and modular, has high-fidelity standards compliance, provides auto-build installation, and has been tested in a relevant context. Support is developer-organized, and an interface guide is available. Software and documentation are applicable for most systems. Brief statements are available describing limited rights for reuse and developers may be contacted to negotiate additional rights.
8	Demonstrated local reusability; the software has been reused by multiple users.	Software has been shown to be extensible and has been qualified through test and demonstration. An extension guide and organization-provided support are available. Brief statements are available describing unrestricted rights for reuse, and developers may be contacted to obtain formal rights statements.
9	Demonstrated extensive reusability; the software is being reused by many classes of users over a wide range of systems.	Software is fully portable and modular, with all appropriate documentation and standards compliance, encapsulated packaging, a GUI installer, and a large support community that provides patches. Software has been tested and validated through successful use of application output. Multiple statements describing unrestricted rights for reuse and the recommended citation are embedded into the product.

4 POTENTIAL USES OF THE REUSE READINESS LEVELS

Capabilities are needed for specifying and verifying component reusability (Frakes & Kang, 2005), and possible uses of reusability measures, such as the RRLs, have been described above and previously (Marshall, Berrick, Bertolli, et al., 2007; Marshall, Downs, & Gerard, 2009). If RRLs are included as metadata for resources in software repositories and catalogs, they can assist software adopters in their efforts to select candidate reusable assets by identifying which assets might be sufficiently ready to meet their reuse needs. While detailed assessments of the selected assets are still necessary before a software solution can be chosen, the availability of criteria and measures for assessing potential reusability can assist in reducing the pool of candidate solutions being considered, allowing adopters to decrease the amount of effort necessary during this stage of the appraisal process. Likewise, the RRLs can assist software producers in their efforts to develop reusable software assets. In particular, the RRL topic area levels can assist producers in identifying areas of their software which, if developed further, could improve the potential for reusing their assets. Also, the sponsors of software production or adoption projects can use the RRLs to assist in establishing requirements for projects to be sponsored, selecting a project to sponsor, and evaluating the selected project and assets developed or adopted during and after the project's period of performance. In the case of data stewardship, RRLs can foster the reuse of software upon which the archived data are dependent for their use and complement existing data reusability measures to assist data stewards in ensuring that their data archives sufficiently address any software dependencies. Potential use cases for software sponsorship, production, adoption, and data stewardship are proposed in Section 5.

Similar to the current use of TRLs, RRLs could be part of contracts that involve the production or adoption of reusable assets. Requests for proposals could ask for explanations of the approach for reusing existing assets or how assets to be developed will be made reusable, and reuse metrics, such as RRLs, could be important for this and for managing the lifecycle of reusable software (Anderson, Morris, Smith, & Ward, 2007). Just as TRL advancement projects may be sponsored now, RRL advancement projects could be sponsored in the future, where existing reusable software assets that have attained a particular RRL rating are developed in such a way as to be released at a higher RRL when the project is completed.

Examples such as these demonstrate uses of the overall RRLs, metrics that may be well-suited for project managers. An RRL assessment consisting of a single numeric value can be understood easily, offering a quick and simple way of measuring the maturation of software assets as they become more reusable. Alternatively, more detailed information may be desired, or even required, by software developers who could then employ the complete nine-by-nine matrix of all the topic area levels offered by the RRLs. As a nine-by-nine matrix, the extent of RRL topic area levels present capabilities for more precise evaluation of software reusability.

Software developers may use the topic area levels to calculate the overall RRL of their software or to improve the reusability of specific aspects of their software, by taking guidance from the topic area levels for further developing specific aspects of the software to enable reuse. Knowledge of the scores for the topic area levels for a reusable asset also could benefit software adopters by providing additional information about its potential reusability. By highlighting the particular aspects in which the software has greater or lesser reusability, the topic area levels can help adopters to improve their estimations of software assets in terms of their potential reuse. Levels for some topic areas also could be used as benchmarks by software developers and adopters when conducting quality assurance. Therefore, if assessments of topic area levels are completed for a particular software product, it would be important to record and report these along with the overall RRL. Such assessments could identify value added from a project, which could help to address the gap between current measurements of software reuse and measurements of the benefits of software reuse described by Dinsoreanu and Ignat (2009).

The potential uses briefly described above offer examples of how RRLs may be applied in different situations to complement existing practices of different classes of users. They also indicate how the presence of information about overall RRLs and about RRL topic area levels can improve capabilities for completing specific reuse activities. The proposed use cases, briefly described below, offer additional details on ways software developers and adopters, their sponsors, and data stewards may make use of the RRLs within their processes.

5 USE CASES FOR THE REUSE READINESS LEVELS

Use cases support many aspects of the development process for a variety of technologies (Jacobson, 2004). In addition, the presentation of use case diagrams along with narrative descriptions enhances understanding of use cases and improves communication (Gemino & Parker, 2009). Proposed sequences of brief use cases and use case diagrams are presented below to describe how the RRLs could contribute to some of the key activities

involving software that are conducted throughout software sponsorship projects, the software development lifecycle, and the data lifecycle. Activities common to traditional and more recent models of the systems development lifecycle include requirements gathering, designing, developing, and testing activities (Guntamukkala, Wen, & Tam, 2006). The use cases demonstrate potential applications of the RRLs to support these activities during software development efforts completed for software production and software adoption. Use case diagrams, graphically depicting the use cases and their actors, are represented using standard Unified Modeling Language (Fowler, 2003).

The following use cases briefly describe the use of the RRLs to support project sponsorship, software production, software adoption, and data stewardship conducted by the organizations that serve as the primary stakeholders engaging in these activities. Initially, the applicability of the RRLs is considered for organizations that sponsor projects which include the production or adoption of potentially reusable software resources. Next, sequences of use cases describe the variety of activities of organizations involved in both the production and the adoption of software that could be supported by the RRLs. One sequence demonstrates uses of the RRLs by software producers who could provide the developed software resource to other developers and users for their potential reuse. A second sequence demonstrates the use of the RRLs to adopt software that is being considered for reuse by potential users of software resources. The last sequence describes the use of the RRLs to complement existing data reusability measures employed by organizations engaged in the stewardship of the data and related software artifacts associated with the data for which they are responsible.

Although presented sequentially within each use case, many of the activities that would be supported by the RRLs could be completed in parallel. In addition, the activities within each use case could be conducted in an iterative manner, which could change the sequences. The RRL use case sequences are presented in Table 2.

Table 2. RRL Use Case Sequences

Case	Project Sponsorship	Software Production	Software Adoption	Data Stewardship
1	Plan Software Project Sponsorship	Plan Software Production	Plan Software Adoption	Establish Data Stewardship Policies
2	Determine Cost of Projects to be Sponsored	Identify Software Requirements	Identify Software Evaluation Criteria	Define Appraisal Criteria
3	Establish Assessment Criteria for Proposals	Identify Software Components	Identify Candidate Software	Negotiate Submission Agreements
4	Identify Candidate Software Proposals	Define Software Deliverables	Assess Software Cost	Review Submissions
5	Evaluate Quality of Software Proposals	Evaluate Software Components	Evaluate Software Components	Package Data
6	Evaluate Progress of Software Project	Evaluate Software Product	Evaluate Software Integration	Maintain Data

In addition to the stakeholder organizations, various roles are reflected in the RRL use cases. Taken together, the roles of individual actors who would use the RRLs during project sponsorship, software production, software adoption, or data stewardship efforts could include program managers, program assistants, potential users, project managers, team leaders, individual software developers, data scientists, data managers, data producers, reviewers, and archivists. Within each use case, the roles of the individual actors who use the RRLs are identified.

5.1 Project Sponsorship Use Case

The project sponsorship use case sequence for the RRLs describes their use by organizations involved in sponsoring projects that include aspects of software production or adoption. Such organizations could include

companies that contract or outsource software development projects and government agencies or foundations, which sponsor projects for their programs that include software development.

Software sponsorship programs enable sponsors and potential users of software to support the production and adoption of software to meet the goals of their programs. The sponsorship of software involves several activities completed by the program manager, the program assistant, and the potential user to establish the requirements for software production or adoption projects, to select among competing projects, and to evaluate awarded projects during the performance period and after completion. Figure 1 presents a use case diagram for project sponsorship.

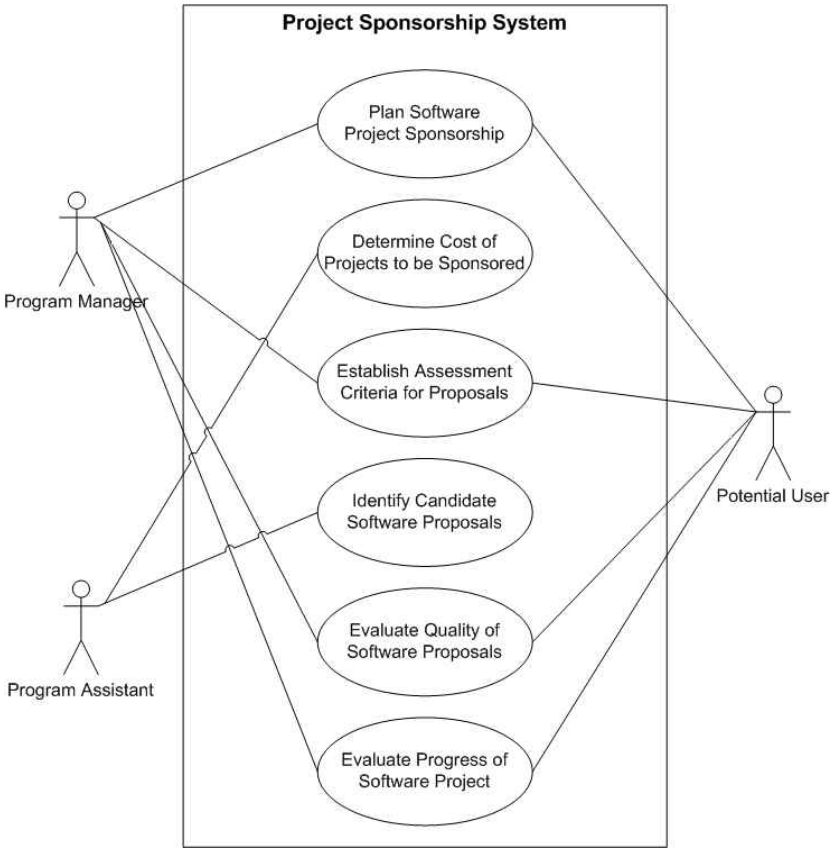

Figure 1. Project Sponsorship Use Case Diagram

5.1.1 Plan Software Project Sponsorship

The program manager and the potential user of the software review the RRL summary levels to identify the requirements for software production or adoption to be met by the project to be sponsored. Each topic area of the RRLs also is considered for inclusion in the specifications to be included in the Request for Proposals (RFP) or Request for Quotations (RFQ) that describe the requirements for proposal submission. The levels within each topic area also are considered to derive requirements for inclusion in the RFP or RFQ. For example, level 9 is selected for Portability.

5.1.2 Determine Cost of Projects to be Sponsored

The selected level for each topic area that has been considered for the RFP or RFQ is evaluated by the program assistant to estimate the costs for attaining each level when developing or adopting software. For Portability, the costs of achieving level 9 are estimated. Lower levels also are evaluated to identify costs of predecessor activities to be conducted to attain the specified level for a topic area. The costs to attain each topic area level are combined, compared, and considered to propose the budget for the program to be sponsored.

5.1.3 Establish Assessment Criteria for Proposals

Descriptions of each required topic area level are reviewed by the program manager and the potential user to identify specific requirements within each topic area to be used as the criteria for assessing software production and adoption proposals received from bidders. Topic area levels that are predecessors of the required topic area levels also are reviewed to identify specific criteria that could be used for assessing competing proposals. Weighting factors are assigned to the criteria that pertain to software reuse so that they can be considered along with other criteria when evaluating proposals.

5.1.4 Identify Candidate Software Proposals

During the initial review of competing proposals received from organizations that bid on the sponsored project, each proposal is reviewed by the program assistant to determine whether proposed deliverables that pertain to software reuse will meet the requirements for the topic area levels specified in the RFP or RFQ. Based on this preliminary review, each proposal receives a weighted score that is considered along with the weighted scores received for criteria that measure whether other specified conditions have been met.

5.1.5 Evaluate Quality of Software Proposals

Each of the competing proposals that have met the conditions of the preliminary reviews are further reviewed by the program manager and the potential user to determine whether the proposed activities specify the delivery of software products that meet the requirements described in the RRLs for the topic area levels specified in the RFP or RFQ. Project proposals that describe the production or adoption of software deliverables that meet the requirements described in each specified topic area level receive further review as contenders for potential sponsorship.

5.1.6 Evaluate Progress of Software Project

The program manager and the potential user review the RRL topic area levels to identify software deliverables to be produced or adopted by the sponsored project. Predecessor topic area levels also are considered for possible inclusion as interim deliverables to be reviewed by the program manager and by the potential user at predetermined points during the project schedule to verify that the project is proceeding as planned. Requirements for critical topic area levels are specified as deliverables associated with scheduled payments. Deliverables received from the project are evaluated iteratively in terms of the specified topic area level requirements to determine whether the project has met the conditions to receive payment. In accordance with the requirements, the software is evaluated to determine whether level 9, for example, has been achieved for Portability.

5.2 Software Production Use Case

Software production could come from companies that develop software products as their primary business or from organizations, institutions, or agencies that develop software to support their core business activities. The production of software involves several activities to ensure that the software meets the functional requirements envisioned for the software and that the software can be used in accordance with such requirements. The following use case activities demonstrate how the RRLs could offer support for various activities completed during the software development process by potential software users, the project manager, team leaders, and individual developers who serve as members of software development teams. A use case diagram for software production is shown in Figure 2.

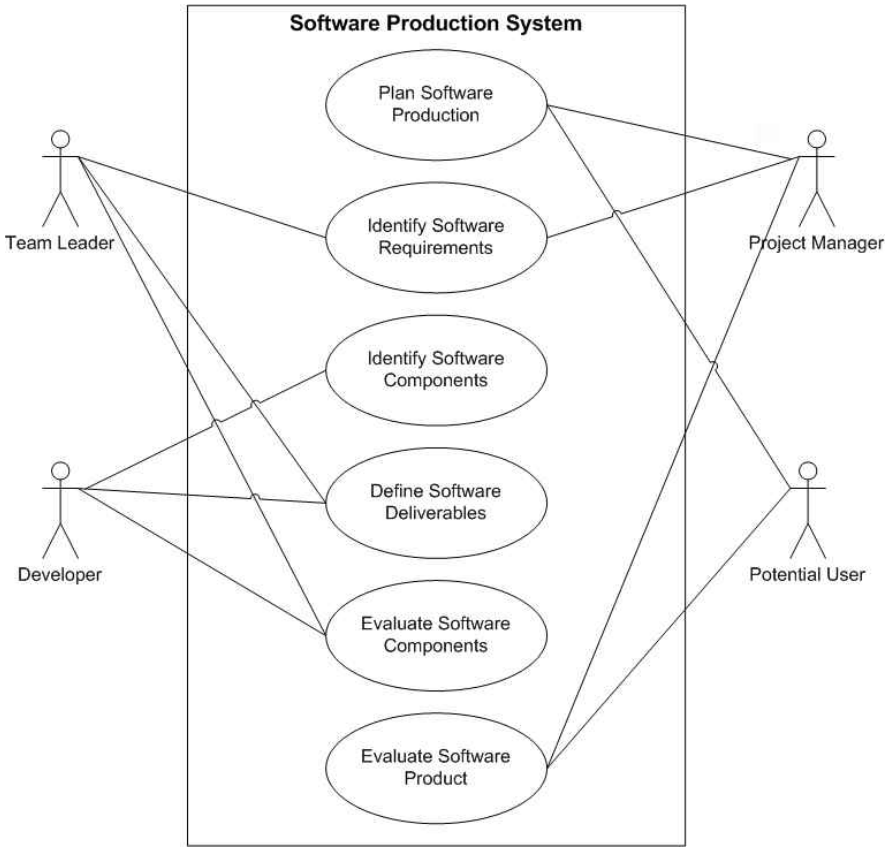

Figure 2. Software Production Use Case Diagram

5.2.1 Plan Software Production

During the initial planning for the creation of a system or software application, the RRL summary levels, topic areas, and topic area levels are reviewed by the software development project manager and the potential user to support the software development project planning activities. They evaluate the potential reusability of the components and products to be produced, including frameworks, technologies, and support considerations, and agree that RRL summary level 8, for example, should be achieved for the software being developed during the project.

5.2.2 Identify Software Requirements

During the software requirements identification process, the topic areas of the RRLs are reviewed by the software development project manager and each development team leader to identify the level that needs to be achieved for each RRL topic area to meet the requirements. They also determine the skills that will be needed throughout the development effort to meet the software requirements and achieve these levels. Based on this review, the team leaders form the teams of developers to focus on each of the requirements that have been identified for the software product.

5.2.3 Identify Software Components

Software components are identified to meet the requirements, and the developer who is assigned to work on a component reads the RRL topic areas to determine which topic areas are pertinent to the assigned development activities. In addition to code modules, examples of software components to be produced could include requirements specifications, frameworks, technologies, documentation, test cases, automatic installers, rights and usage statements, and the support system. The developer refers to the appropriate levels for each relevant topic area to determine the steps that need to be completed for each component being developed. For example, the RRLs for documentation are consulted by the developer assigned to develop the documentation for the planned software. The technical evaluation of components includes the use of frameworks, technologies, and the amount of support to be provided for the software being produced.

5.2.4 Define Software Deliverables

The RRLs for each topic area are used by each development team leader and each developer to define and agree upon intermediate results or deliverables to be completed by the assigned developer for inclusion in the development plan. For example, each of the deliverables for developing the software documentation reflects a level of the RRLs for documentation.

5.2.5 Evaluate Software Components

The software development team leader and the developer compare completed development assignments with the targeted level of the respective RRL topic area to ensure that the requirements for that level have been met. For example, the software documentation is compared to the RRL documentation level requirements to determine whether the software documentation has attained that level of completion. Similarly, frameworks, technologies, and planned levels of support are compared across RRL topic areas to ensure that RRL summary level 8, for example, has been achieved.

5.2.6 Evaluate Software Product

After the components have been assembled and tested, the assembled software product is assessed by the software development project manager and the potential user to determine the status of the software product delivered by comparing the RRL summary level description with the software that was produced.

5.3 Software Adoption Use Case

Commercial firms, non-profit organizations, educational institutions, and government agencies might decide to adopt software that has been created by others rather than to develop their own. The adoption of software resources involves several activities to identify, select, and prepare software resources for potential reuse. The use of the RRLs could apply in situations where a major system developer is preparing a proposal for or has won a competed contract for system development and is considering whether to make, buy, or reuse software. The RRLs can offer support for various activities that are conducted by the software development project manager, team leaders, potential users, and software developers to ensure that the software can be reused while attaining the potential benefits, such as quality improvements and cost savings, made possible by adopting software resources that have been developed initially for use within another context. Figure 3 depicts a use case diagram for software adoption.

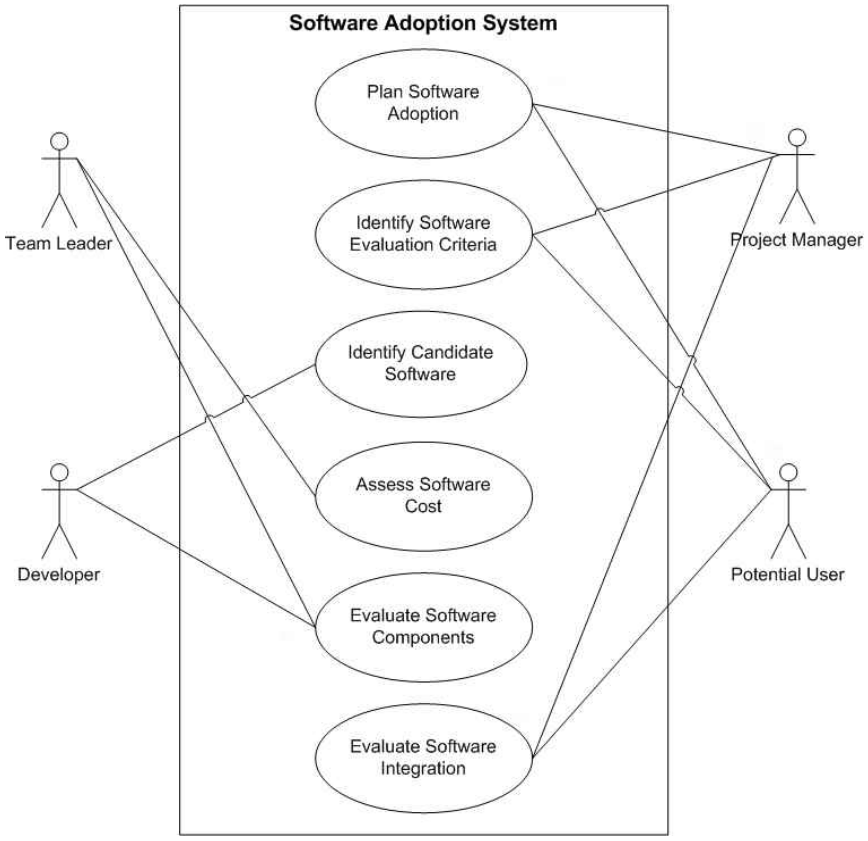

Figure 3. Software Adoption Use Case Diagram

5.3.1 Plan Software Adoption

The software development project manager and the potential user assess the RRL summary levels, topic areas, and topic area levels to determine how these can be used to guide the decision to build or adopt components for a planned software development project. They determine the specific RRL topic area levels that must be inherent in the software that is needed by the potential user. Based on subsequent analysis, they decide to adopt software and to use the RRL levels to determine project milestones and associated deliverables.

5.3.2 Identify Software Evaluation Criteria

The software development project manager and the potential user refer to the RRL topic area levels to identify the criteria for assessing candidate software for potential adoption. These criteria can then be used later to determine whether the effort of adopting candidate software components and the reuse framework would be cost-effective for the current software development project. After reviewing the RRLs, they decide that software and related components that have achieved RRL summary level 8, for example, could be considered for potential adoption.

5.3.3 Identify Candidate Software

The software developer identifies potential software candidates by comparing each software component and the reuse framework to the targeted RRL topic area level to determine whether it has reached level 8, which was previously targeted. For some software components that have not reached level 8 in some topic areas, the developer assesses how much additional development is required for the deficient topic areas.

5.3.4 Assess Software Cost

The software development team leader compares the state of each candidate software component and the reuse framework to the targeted RRL topic area levels. Additional development activities are identified that would need to be completed for each component, technology, and planned level of support to achieve the minimum acceptable level for each topic area. The cost of these development activities is compared to that of new

development, which also is assessed with the RRL topic area levels, to determine the more cost-effective solution.

5.3.5 Evaluate Software Components

The software development team leader and individual developers evaluate the quality of the software components and reuse framework that has been designated for adoption by comparing the state of each software component against the RRL topic area levels. Each developer is assigned to complete components, technology, and planned levels of support that require additional development, and the development team leader and the developer refer to the topic area levels to assess progress on the assigned components and reuse framework.

5.3.6 Evaluate Software Integration

The software development project manager and the potential user assess progress on the adopted software by comparing the customized and integrated components and reuse framework with RRL summary level 8, which was expected to be achieved according to the schedule. Additional development and integration activities are targeted to the components and reuse framework, technology, and planned level of support that have been identified as deficient.

5.4 Data Stewardship Use Case

Organizations involved in the stewardship of data could include scientific data archives and other archives, data centers, research institutes, research libraries, museums, institutional repositories, and data repositories. These and other data stewardship organizations manage data to facilitate current and future use of the data they receive and are referred to as archives within this sequence of use case activities.

Supporting data stewardship, the RRLs can complement existing data reusability measures employed for the management, preservation, and dissemination of data ingested in archives. Contingent upon the data types represented within the data, the RRLs can be used along with current data reusability practices to evaluate and facilitate the reuse of software that present dependencies for the data being preserved for future use. When archiving data and other scientific or technical information, which could include software and related artifacts that have been developed to either generate or render the data, use of the RRLs by data managers, data scientists, data producers, reviewers, and archivists can improve capabilities for planning, appraisal, preparation, review, packaging, and maintenance activities. The data stewardship use case is represented as a use case diagram in Figure 4.

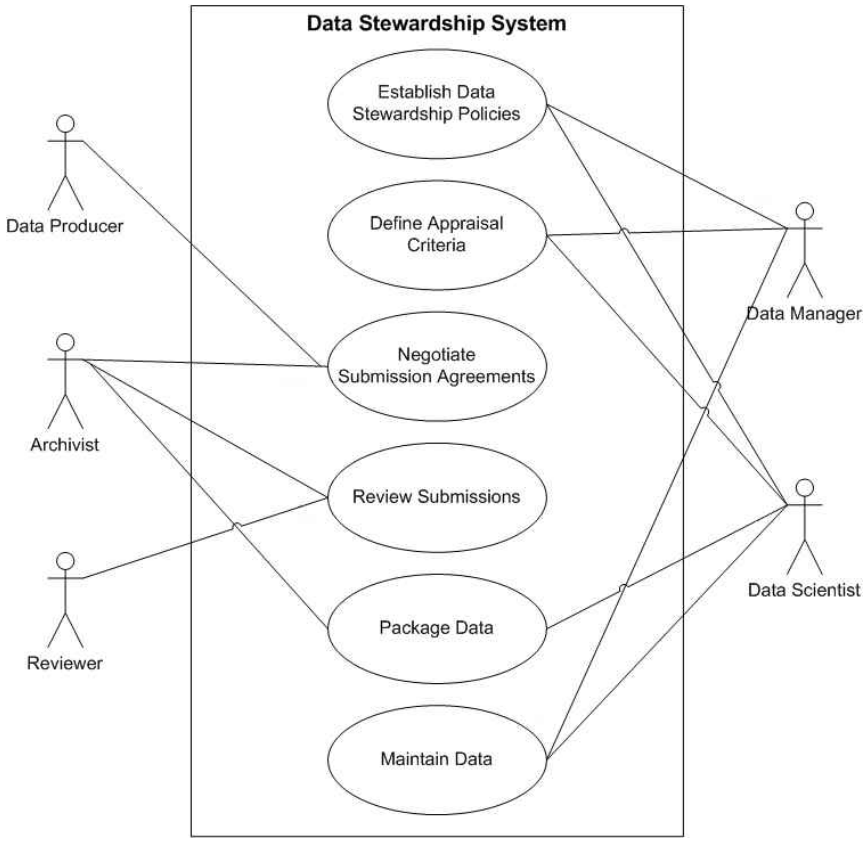

Figure 4. Data Stewardship Use Case Diagram

5.4.1 Establish Data Stewardship Policies

When planning to create data management and archival programs, data managers and data scientists use the RRL summary levels and the RRL topic areas to identify quality assurance objectives to be met when establishing archival processes for collections of digital data that have software dependencies. The selection of specific levels and topic areas for each collection would depend on its mission, purpose, and collection development plan.

5.4.2 Define Appraisal Criteria

Data managers and data scientists use the RRL topic areas to establish some of the criteria for appraising data sets that are candidates for accession into the collections of data that they manage. If necessary, minimum levels of acceptance are established for topic areas considered critical for enabling data stewardship of data that are dependent on software. The data manager and the data scientist agree that achieving RRL summary level 9, for example, is necessary for the software artifacts that pose dependencies for data sets that are planned for acquisition by the archive.

5.4.3 Negotiate Submission Agreements

Specific levels within each RRL topic area are referenced mutually by the archivist and the data producer when negotiating the data submission agreement to identify aspects of content and representation information to be included in the submission information package of software-dependent data that are being considered for transfer to the archive. The archive requests identification and inclusion, within the submission information package, of software and system components that present dependencies for the data, including the reuse framework, technology, and required level of support for the software. The archive also requests that the software included in the submission information package for software-dependent data have met the criteria for RRL summary level 9, for example.

5.4.4 Review Submissions

During the review of the submission information package received, the reviewer and the archivist consult specific RRL topic area levels to determine whether the submission of software-dependent data has been prepared in accordance with the levels specified within the submission agreement established for the data being reviewed. After determining that the software is not sufficiently documented, for example, additional documentation is requested.

5.4.5 Package Data

The RRL topic areas and levels within each topic area are used by the archivist and the data scientist to identify the information about software that needs to be contained either within dissemination information packages being created to distribute data for use by designated communities of users or within submission information packages being created to transfer the data to other facilities.

5.4.6 Maintain Data

The RRL topic area levels are reviewed by the data manager and the data scientist to identify the requirements for computer programs and scripts that will be created for converting data from legacy formats to current formats that will be supported by software or emulate software that either were used to generate the data or will be used to render the data in the future. The software used to generate data and render data also are reviewed periodically, in terms of the RRLs, as part of the ongoing technology assessment.

6 SUMMARY

Developments in capabilities for data stewardship and data interoperability are enabling new uses of scientific data and research-related information. These capabilities offer opportunities for data integration and analysis that may have been unforeseen by the original data collectors. Software and systems often facilitate these new opportunities for using data, resulting in dependencies upon the software to use the data. Software dependencies present challenges for the future use of data but can be addressed by identifying the software that present dependencies for data, preparing such software for reuse, and by evaluating the potential reusability of software to guide current and future decisions about software reuse and about the use of data that are dependent upon software. Instruments for measuring the reusability of software can contribute to capabilities for assessing software and for evaluating the results of activities completed during the software development lifecycle and the data lifecycle.

The Reuse Readiness Levels (RRLs) offer capabilities for measuring the maturity of system components and reuse frameworks for potential reuse. Use cases demonstrate how use of the RRLs can contribute to efforts that sponsor software production and adoption, to development efforts that produce software or system components, to development efforts that adopt software or system components, and to efforts that archive data and software when conducting data stewardship activities. We hope that actual uses and applications of the RRLs extend beyond those currently envisioned to provide additional benefits for software sponsors, software adopters, software producers, and data stewards as well as for others. We also hope that use of the RRLs can contribute to their further improvement to become valuable measures of the potential reusability of software and of the activities that require the measurement of software reusability.

7 CONCLUSION

As the uses of scientific data evolve over time, so must the related technologies upon which they depend. The reliance on operating systems, software applications, and other software artifacts becomes increasingly important as new data products and services are developed for using and integrating data. In consideration of the dependence on software artifacts that is experienced by current users of scientific data products and services, the reusability of software becomes a critical aspect of managing the software upon which the future use of the data is dependent. Attaining capabilities for the management and stewardship of scientific data also includes obtaining the tools and instruments for managing the software artifacts that are needed to use and maintain the data. Developing better tools to facilitate the reusability of software is an important step in ensuring that required software can be reused as technologies change and evolve.

Capabilities for measuring the potential reusability of software and related artifacts can contribute to the preparation, management, and stewardship of scientific data. Developing tools that can measure the reusability

of software can assist sponsors of software development projects, software producers, software adopters, and data stewards in their efforts to ensure that software components can be reused easily in the future. Using reusability measurement tools, such as the Reuse Readiness Levels, offers the potential for these and other users to assess the potential reusability of the systems and software artifacts that are being relied upon to perform today and in the future.

8 ACKNOWLEDGEMENTS

The authors appreciate the contributions of the current and previous members of the NASA Earth Science Data Systems Software Reuse Working Group on the development of the Reuse Readiness Levels presented here, including suggestions for improvement of a previous draft of this paper by Christine Whalen of INNOVIM. The authors also are grateful for the constructive comments for improving the paper that were received from the referees and for the support received from the NASA for the work reported in this paper, including the support for Robert Downs under Contract NNG08HZ11C. The opinions expressed here are those of the authors and do not necessarily reflect the opinions of their employers or NASA.

9 REFERENCES

Agosti, M., Di Nunzio, G.M., & Ferro, N. (2007). The Importance of Scientific Data Curation for Evaluation Campaigns. In Thanos, C., Borri, F., & Candela, L. (Eds.) *Digital Libraries: Research and Development, First International DELOS Conference*, Pisa, Italy, February 13–14, 2007, Revised Selected Papers, New York, Springer, 157–166.

Albani S., & Giaretta D. (2009) Long-term Preservation of Earth Observation Data and Knowledge in ESA through CASPAR. *International Journal of Digital Curation* 4(3), 4–16. Retrieved May 15, 2010 from the World Wide Web: http://www.ijdc.net/index.php/ijdc/article/view/130

Anderson W. L. (2004) Some challenges and issues in managing, and preserving access to, long-lived collections of digital scientific and technical data. *Data Science Journal* 3, 191–201. Retrieved May 15, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/3/0/3_191/_article

Anderson, W., Morris, E., Smith, D., & Ward, M.C. (2007) *COTS and Reusable Software Management Planning: A Template for Life-Cycle Management*. Carnegie Mellon University, Software Engineering Institute, Technical Report CMU/SEI-2007-TR-011. Retrieved November 24, 2009 from the World Wide Web: http://www.sei.cmu.edu/reports/07tr011.pdf

Barton, C., Smith, R., & Weaver, R. (2010) Data Practices, Policy, and Rewards in the Information Era Demand a New Paradigm. *Data Science Journal* 9, IGY95–IGY99. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/9/0/9_IGY95/_article

Berman, F. (2008) Got data?: A Guide to Data Preservation in the Information Age. *Communications of the ACM 51*(12), 50–56.

Cavalcanti, M. C., Mattoso, M., Campos, M. L., Llirbat F., & Simon, E. (2002). Sharing Scientific Models in Environmental Applications. *Proceedings of the 2002 ACM symposium on Applied computing*, Madrid, Spain (pp. 453–457). Retrieved November 24, 2009 from the World Wide Web: http://portal.acm.org/

Cheney, J., Ahmed, A., & Acar, U. A. (2007) Provenance as Dependency Analysis. In: *Database Programming Languages*, Berlin / Heidelberg: Springer.

Consultative Committee for Space Data Systems (2002) Reference Model for an Open Archival Information System (OAIS). Adopted as: Space data and information transfer systems - Open archival information system - Reference model (ISO 14721:2003). Retrieved May 15, 2010 from the World Wide Web: http://public.ccsds.org/publications/archive/650x0b1.pdf

Department of Defense (2009) Technology Readiness Assessment (TRA) Deskbook. Retrieved November 24, 2009 from the World Wide Web: http://www.dod.mil/ddre/doc/DoD_TRA_July_2009_Read_Version.pdf

Dinsoreanu, M., & Ignat, I. (2009) A Pragmatic Analysis Model for Software Reuse. In: R. Lee & N. Ishii (Eds.): *Sofware Engineering Research, Management & Applications 2009*, Berlin: Springer-Verlag.

Downs, R. R., & Chen, R. S. (2010) Designing submission and workflow services for preserving interdisciplinary scientific data. *Earth Science Informatics*. DOI: 10.1007/s12145-010-0051-6

Dubin, D., Futrelle, J., Plutchak, J., & Eke, J. (2009) Preserving Meaning, Not Just Objects: Semantics and Digital Preservation. *Library Trends* 57(3), 595–610. DOI: 10.1353/lib.0.0054

Fowler, M. (2003) *UML Distilled: A Brief Guide to the Standard Object Modeling Language, Third Edition*, Reading, MA: Addison Wesley Professional.

Fox, P., McGuinness D. L., Cinquini L., West P., Garcia J., Benedict J. L., & Middleton, D. (2009) Ontology-Supported Scientific Data Frameworks: The Virtual Solar-Terrestrial Observatory Experience. *Computers & Geosciences* 35, 724–738.

Frakes, W. B., & Kang, K. (2005) Software Reuse Research: Status and Future. *IEEE Trans. Software Eng. 31*(7), 529–536.

Gemino, A., & Parker, D. (2009). Use Case Diagrams in Support of Use Case Modeling: Deriving Understanding from the Picture. *Journal of Database Management, 20*(1), 1–24.

Gold, R., & Jakubek, D. (2005) Technology Readiness Assessments for IT and IT-Enabled Systems, *CrossTalk.* May 2005, 25–30.

Graettinger, C.P., Garcia, S., Siviy, J., Schenk, R.J., & Van Syckle, P.J. (2002) *Using the Technology Readiness Levels Scale to Support Technology Management in the DoD's ATD/STO Environments*. Carnegie Mellon University, Software Engineering Institute, Special Report CMU/SEI-2002-SR-027. Retrieved November 24, 2009 from the World Wide Web: http://www.sei.cmu.edu/reports/02sr027.pdf

Guntamukkala, V., Wen, H. J., & Tarn, J. M. (2006) An empirical study of selecting software development life cycle models. *Human Systems Management 25*(4), 265–278.

Hill. L., Crosier, J., Smith. T., & Goodchild, M. (2001) A content standard for computational models. *D-Lib Magazine 7*(6). Retrieved November 24, 2009 from the World Wide Web: http://www.dlib.org/dlib/june01/hill/06hill.html

Hu, C., Ouyang, C., Wu, J., Zhang, X., & Zhao C. (2009). Non-Structured Materials Science Data Sharing Based on Semantic Annotation. *Data Science Journal* 8, 52–61. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/8/0/8_52/_article

Hunter, J. (2006) Scientific Publication Packages – A Selective Approach to the Communication and Archival of Scientific Output. *The International Journal of Digital Curation 1*(1), 33–52.

Interagency Working Group on Digital Data (2009) *Harnessing the Power of Digital Data for Science and Society*. Report of the Interagency Working Group on Digital Data to the Committee on Science of the National Science and Technology Council, January, 2009. Washington, DC: National Science and Technology Council, Committee on Science. Retrieved May 15, 2010 from the World Wide Web: http://handle.dtic.mil/100.2/ADA501489

Jacobs, J. A., & Humphrey, C. (2004) Preserving Research Data. *Communications of the ACM* 47(9), 27–29.

Jacobson, I. (2004) Use cases – Yesterday, Today, and Tomorrow. *Software and Systems Modeling, 3*(3), 210–220.

Janée, G., Frew, J., & Moore, T. (2009) Relay-supporting Archives: Requirements and Progress. International *Journal of Digital Curation* 4(1) 57–70. Retrieved May 15, 2010 from the World Wide Web: http://www.ijdc.net/index.php/ijdc/article/view/102

Knight, G., & Pennock, M. (2009) Data Without Meaning: Establishing the Significant Properties of Digital Research. *The International Journal of Digital Curation* 4(1) 159–174. Retrieved May 15, 2010 from the World Wide Web: http://ijdc.net/index.php/ijdc/article/view/110/0

Kühne, M., & Liehr, A. W. (2009) Improving the Traditional Information Management in Natural Sciences. *Data Science Journal* 8, 8–26. Retrieved May 15, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/8/0/8_18/_article

Lynch, C. (2008) Big data: How do your data grow? *Nature 455*, 28–29.

Mankins, J.C. (1995) *Technology Readiness Levels: A White Paper*. Retrieved November 24, 2009 from the World Wide Web: http://www.hq.nasa.gov/office/codeq/trl/trl.pdf

Marshall, J.J., Berrick, S.W., Bertolli, A., Burrows, H., Delnore, V.E., Downs, R.R., Enloe, Y., Falke, S., Folk, M., Gerard, N., Gerard, R., Hunter, M., Jasmin, T., McComas, D., Samadi, S., Sherman, M., Swick, R., Tilmes, C., & Wolfe, R.E. (2007) A Community-Developed Measurement of the Reusability of Software Through Reuse Readiness Levels. *Eos Trans. AGU. 88*(52) Fall Meet. Suppl., Abstract IN31A-0074.

Marshall, J.J., & Downs, R.R. (2008) Reuse Readiness Levels as a Measure of Software Reusability. *Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International, 3*, pp. III-1414–III-1417.

Marshall J.J., Downs R.R., & Gerard, R.S. (2009) Measuring Software Reusability for Scientific Data Systems. *Eos Trans. AGU 90*(52), Fall Meet. Suppl.. Abstract IN11C-1061.

Marshall J.J., Downs, R.R., & Samadi, S. (2010) Building the Next Generation of Aerospace Data Processing Systems by Reusing Existing Software Components. In Arif, T.T., (Ed.), *Aerospace Technologies Advancements*. Croatia: IN-TECH

Marshall, J.J., Downs, R.R., Samadi, S., Gerard, N.S., & Wolfe, R.E. (2008) Software Reuse to Support Earth Science. *Journal of Frontiers of Computer Science and Technology 2*(3), 296–310.

Marshall, J.J, Olding, S.W., Wolfe, R.E., & Delnore, V.E. (2006) Software Reuse Within the Earth Science Community. *Geoscience and Remote Sensing Symposium, 2006. IGARSS 2006. IEEE International Conference on*, 2880–2883.

Matthews, B., Shaon, A., Bicarregui, J., Jones, C., Woodcock, J., & Conway, E. (2009) Towards a Methodology for Software Preservation. *Proceedings of iPRES 2009: the Sixth International Conference on Preservation of Digital Objects*. California Digital Library. University of California. 132–140. Retrieved May 15, 2010 from the World Wide Web: http://escholarship.org/uc/item/8089m1v1

McGuinness, D.L., Fox, P., Cinquini, L., West, P., Garcia, J., Benedict, J.L., & Middleton, D. (2008). Enabling Scientific Research Using an Interdisciplinary Virtual Observatory: The Virtual Solar-Terrestrial Observatory Example. *AI Magazine, 29*(1), 65–76.

Moreau, L., Clifford, B., Freire, J., Gil, Y., Groth, P., Futrelle, J., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Simmhan, Y., Eric Stephan & Van den Bussche, J. (2009) Open Provenance Model Core Specification v1.1. Technical Report, University of Southampton. Retrieved May 22, 2010 from the World Wide Web: http://eprints.ecs.soton.ac.uk/18332/

Mohagheghi, P., & Conradi, R. (2007) Quality, productivity and economic benefits of software reuse: a review of industrial studies. *Empirical Softw. Engg. 12*(5), 471–516.

National Aeronautics and Space Administration (2008a) 2008 NASA Software of the Year Summary Evaluation Document. Retrieved November 24, 2009 from the World Wide Web: http://www.nasa.gov/pdf/251088main_2008_SWOY_Sum_Eval_Doc and Defs.pdf

National Aeronautics and Space Administration (2008b) NASA Research and Technology Program and Project Management Requirements, NPR 71280.8. Retrieved November 24, 2009 from the World Wide Web: http://nodis3.gsfc.nasa.gov/displayDir.cfm?t=NPR&c=7120&s=8

NASA ESDS Software Reuse WG (2007a) Reuse Readiness Levels (RRLs) – A Work in Progress. Retrieved November 24, 2009 from the World Wide Web: http://www.esdswg.org/softwarereuse/Resources/library/6th-esds-wg-meeting/ESDSWG6RRLSlides.pdf/view

NASA ESDS Software Reuse WG (2007b) Reuse Readiness Levels (RRLs): Proposed Topic Area Levels. Retrieved November 24, 2009from the World Wide Web: http://www.esdswg.org/softwarereuse/Resources/library/6th-esds-wg-meeting/ESDSWG6RRLPoster.ppt/view

NASA ESDS Software Reuse WG (2008a) Reuse Readiness Levels (RRLs) – A Work in Progress. Retrieved November 24, 2009 from the World Wide Web: http://www.esdswg.org/softwarereuse/Resources/events/event_highlights/2008_Winter_ESIP-RRL-Slides.pdf/view

NASA ESDS Software Reuse WG (2008b) Updated Draft Reuse Readiness Levels (RRLs). Retrieved November 24, 2009 from the World Wide Web: http://www.esdswg.com/softwarereuse/Resources/library/7th-esds-wg-meeting/RRL_Descriptions_v10-esdswg7.pdf/view

NASA ESDS Software Reuse Working Group (2010). Reuse Readiness Levels (RRLs), Version 1.0. April 30, 2010. Retrieved April 30, 2010 from the World Wide Web: http://www.esdswg.org/softwarereuse/Resources/rrls/

National Science Board (2005) Long-Lived Digital Data Collections: Enabling Research and Education in the 21st Century. Washington, DC: National Science Foundation. Retrieved May 15, 2010 from the World Wide Web: http://www.nsf.gov/pubs/2005/nsb0540/nsb0540.pdf

National Science Foundation Cyberinfrastructure Council. (2007). Cyberinfrastructure Vision for 21st Century Discovery. Washington, DC: National Science Foundation.

Ojala, T., & Over, H. H. (2009) Approaches in Using MatML As a Common Language for Materials Data Exchange. *Data Science Journal* 7, 179–195. Retrieved May 15, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/7/0/7_179/_article

OpenBRR.org (2005) Business Readiness Rating for Open Source: Proposed Open Standard to Facilitate Assessment and Adoption of Open Source Software. Retrieved December 2, 2009 from the World Wide Web: http://www.openbrr.org/wiki/images/d/da/BRR_whitepaper_2005RFC1.pdf

OPEN Process Framework (2005) Technology Readiness Assessment. Retrieved November 24, 2009 from the World Wide Web: http://www.opfro.org/Components/WorkProducts/ArchitectureSet/TechnologyReadinessAssessment/TechnologyReadinessAssessment.html

Peterson, W. K. (2009) Open Access to Digital Information: Opportunities and Challenges Identified During the Electronic Geophysical Year. *Data Science Journal* 6, S108–S112. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/8/0/8_S108/_article

PREMIS Editorial Committee (2008) PREMIS Data Dictionary for Preservation Metadata, version 2.0. Retrieved May 22, 2010 from the World Wide Web: http://www.loc.gov/standards/premis/v2/premis-2-0.pdf

Rajasekar, A., Moore, R., Wan, M., & Schroeder, W. (2010) Policy-based Distributed Data Management Systems. *Journal of Digital Information* 11(1). Retrieved May 16, 2010 from the World Wide Web: http://journals.tdl.org/jodi/article/view/756

Sadin, S.R., Povinelli, F.P., & Rosen, R. (1989) The NASA technology push towards future space mission systems, *Acta Astronautica*, *20*, 73–77.

Sharma, A., Grover, P.S., & Kumar, R. (2009) Reusability assessment for software components. *SIGSOFT Softw. Eng. Notes 34*(2), 1–6.

Smith, J. (2004a) An Alternative to Technology Readiness Levels for Non-Developmental Item (NDI) Software. Carnegie Mellon University, Software Engineering Institute, Technical Report CMU/SEI-2004-TR-013. Retrieved November 24, 2009 from the World Wide Web: http://www.sei.cmu.edu/reports/04tr013.pdf

Smith, J.D., II. (2004b) ImpACT: An Alternative to Technology Readiness Levels for Commercial-Off-The-Shelf (COTS) Software. In Kazman, R., & Port D., (Eds.), *COTS-Based Software Systems*, Berlin / Heidelberg: Springer-Verlag.

Smith, J.D., II (2005) An Alternative to Technology Readiness Levels for Non-Developmental Item (NDI) Software. *HICSS '05: Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, Big Island, USA.

Thieman, J. R., Roberts, D. A., King, T. A., Harvey, C. C., Perry, C. H., & Richards, P. J. (2010) SPASE and the Heliophysics Virtual Observatories. *Data Science Journal* 9, IGY85–IGY93. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/9/0/9_IGY85/_article

Tzitzikas, Y. (2009) On Providing Intelligibility-Aware Preservation Services for Digital Objects. *Data Science Journal* 8, 139–151. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/8/0/8_139/_article

To Stand the Test of Time: Long-term Stewardship of Digital Data Sets in Science and Engineering. (2006). A report to the National Science Foundation from the ARL Workshop on New Collaborative Relationships: The Role of Academic Libraries in the Digital Data Universe, September 26–27, 2006, Arlington VA. Retrieved May 16, 2010 from the World Wide Web: http://www.arl.org/bm~doc/digdatarpt.pdf

Uhlir, P. F., Chen, R. S., Gabrynowicz, J. I., & Janssen, K. (2009). Toward Implementation of the Global Earth Observation System of Systems Data Sharing Principles. *Data Science Journal* 8, GEO1–GEO91. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/8/0/8_GEO1/_article

Woods, K., & Brown, G. (2008) Migration Performance for Legacy Data Access. *International Journal of Digital Curation* Vol 3(2), 74–88. Retrieved May 15, 2010 from the World Wide Web: http://www.ijdc.net/index.php/ijdc/article/view/88

Yamagishi, Y., Nagao, H., Suzuki, K., Tamura, H., Hatakeyama, T., Yanaka H., & Tuboi S. (2009) Google Earth as Geoscience Data Browser Project: Development of a Tool to Convert JAMSTEC Research Vessel Navigation Data to KML. *Data Science Journal* 8, S85–S91. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/8/0/8_S85/_article

Zhang, X., Hu, C., & Li, H. (2009) Semantic Query on Materials Data Based on Mapping MatML to an OWL Ontology. *Data Science Journal* 8, 1–17. Retrieved May 16, 2010 from the World Wide Web: http://www.jstage.jst.go.jp/article/dsj/8/0/8_1/_article

(Article history: Received 23 February 2009, Accepted 12 July 2010, Available online 17 July 2010)