AFRL-RI-RS-TR-2009-260
Final Technical Report
November 2009

SOFTWARE SYSTEMS STOCKROOM

The Boeing Company

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW, Wright-Patterson AFB Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2009-260 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/
GEORGE RAMSEYER
Work Unit Manager

/s/
EDWARD J. JONES, Deputy Chief
Advanced Computing Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
NOVEMBER 2009	Final	February 2009 – August 2009

4. TITLE AND SUBTITLE

SOFTWARE SYSTEMS STOCKROOM

5a. CONTRACT NUMBER
FA8750-09-C-0026

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
63781D

6. AUTHOR(S)

James L. Paunicka, Douglas A. Stuart, Andrew M. Vandivort, Gabor Karsai, and Christopher P. VanBuskirk

5d. PROJECT NUMBER
SSTT

5e. TASK NUMBER
BG

5f. WORK UNIT NUMBER
09

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The Boeing Company
Boeing Research & Technology
P.O. Box 516
St. Louis, MO 63166-0516

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RITA
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
N/A

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2009-260

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# 88ABW-2009-4792 Date Cleared: 18-November-2009

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This final report contains a record of activities and a description of project deliverables and results from the Software Systems Stockroom (S3) Phase 1 program, which was executed by Boeing Research & Technology (BR&T) form 05-February-2009 to 02-August-2009. BR&T subcontractors on the effort included Raytheon and Vanderbilt University. The work included development of an open, community-driven, technically focused shared infrastructure that will encourage the capture and reuse of domain knowledge related to development of autonomous flight systems embedded software on emerging processing systems. The domain of autonomous flight systems extends to that of unmanned aerial vehicles (UAVs) and intelligent weapons. The S3 infrastructure includes support for capture and use of Department of Defense domain knowledge; community and information sharing; repositories for components, models, architectures and platforms; support for provenance and version control; and domain-specific taxonomies and ontologies to support user interface generation and rich searching capabilities. The work also includes considerable community engagement and development of candidate Phase 2 plans for S3 development and operation.

15. SUBJECT TERMS
Software Systems Stockroom, Unmanned Aerial Vehicle (UAV), Embedded Flight Software, Domain Knowledge, Intelligent Weapons, Repository, Ontologies

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			George O. Ramseyer
U	U	U	UU	107	19b. TELEPHONE NUMBER (*Include area code*) N/A

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

1.0 SUMMARY

The work reported herein was performed by a team led by the Boeing Company under Air Force Research Laboratory (AFRL) Contract FA8750-09-C-0026, "Software Systems Stockroom (S3)." The work was performed for the AFRL Information Directorate, Advanced Computing Division, Computing Technology Applications Branch (AFRL/RITB). Other government stakeholders involved in this program included the Office of the Secretary of Defense (OSD), the Army Research Laboratory (ARL), the Office of Naval Research (ONR), the Naval Research Laboratory (NRL), and consultants from the Software Engineering Institute (SEI). The Boeing-led team included Raytheon and Vanderbilt University.

The overall objective of this S3 effort was to develop an open, community-driven, technically focused shared infrastructure that will encourage the capture and reuse of domain knowledge related to the development of autonomous flight systems embedded software on emerging processing systems. The S3 infrastructure includes support for the capture and the use of Department of Defense (DoD) domain knowledge; community collaboration and information sharing; repositories for components, models, architectures and platforms; support for provenance and version control; and domain-specific taxonomies and ontologies to support user interface generation and rich searching capabilities.

The domain of autonomous flight systems includes unmanned aerial vehicles (UAVs) and smart weapons, and was chosen as the domain to support for a number of reasons. UAVs and smart weapons represent a DoD growth area within all the armed services. Hosting a mix of on-board vehicle management and mission management (including functions that were performed by aircrew on manned platforms) functions on these platforms provides significant software producibility challenges. In terms of building and serving a wide community associated with this domain, there is significant interest in this area from members of the mil-aero industry, as well as other associated industries (embedded software providers, tools developers, etc.), small business, and academia. Aspects of the supported domain associated with hosting embedded flight software on emerging processing systems (e.g., multi-core) are also significant. With more on-board capabilities desired on aircraft, this element of the S3 domain can provide important support for those needing help solving multi-core development issues.

The Phase 1 work described in this report includes the activities involved in the development of the prototype S3, in populating the prototype with an initial set of artifact contents, and in engaging a community of domain knowledge suppliers and users working this domain. Also included in this report is a description of the as-built capabilities that have been implemented in the S3 prototype repository and the features inherent in the S3 repository contents that have been populated in Phase 1.

The Phase 1 work has produced an S3 with rich repository support functions and populated with contents useful to the domain. A diverse community has been defined and in most cases has already engaged for participation in S3. Moving into a Phase 2 effort for development and operation of S3 will provide an important resource for this community to support affordable development of embedded software for the growing DoD area of autonomous flight systems.

2.0 INTRODUCTION

The subject, the purpose and the scope of this effort are presented in this section.

2.1. Subject

The work reported herein was performed by a team led by the Boeing Company under Air Force Research Laboratory Contract FA8750-09-C-0026, "Software Systems Stockroom (S3) Phase 1." The work was performed for AFRL/RITB. The AFRL Contract Technical Representative was Dr. George Ramseyer, of AFRL/RITB. Additional technical coordination was done with Steve Drager and Bill McKeever, also from AFRL/RITB. Other government stakeholders involved in this program included OSD, ARL, ONR, NRL, and consultants from SEI.

The Boeing-led team included Raytheon and Vanderbilt University. For this effort the Boeing Program Manager was Patrick J. Stokes, the Principal Investigator (PI) was Dr. James L. Paunicka, and the contributing Research & Development Engineers were Dr. Douglas Stuart of the Boeing Company, Andrew Vandivort of Raytheon, and Prof. Gabor Karsai and Chris Van Buskirk of Vanderbilt University. All of the Boeing performers on the contract are part of the Boeing Research & Technology (BR&T) organization. BR&T is the central research, development, and innovation organization for Boeing, chartered with creating the future of aerospace for both the Integrated Defense Systems (defense and space) and Boeing Commercial Aircraft business units.

2.2. Purpose

The overall objective of this S3 effort was to develop an open, community-driven, technically focused shared infrastructure that will encourage the capture and reuse of domain knowledge related to the development of embedded autonomous systems flight software on emerging processing systems. The S3 infrastructure includes support for the capture and the use of Department of Defense domain knowledge; community collaboration and information sharing; repositories for components, models, architectures and platforms; support for provenance and version control; and domain-specific taxonomies and ontologies to support user interface generation and rich searching capabilities. The relationships between key concepts associated with this S3 effort are illustrated in Figure 1.

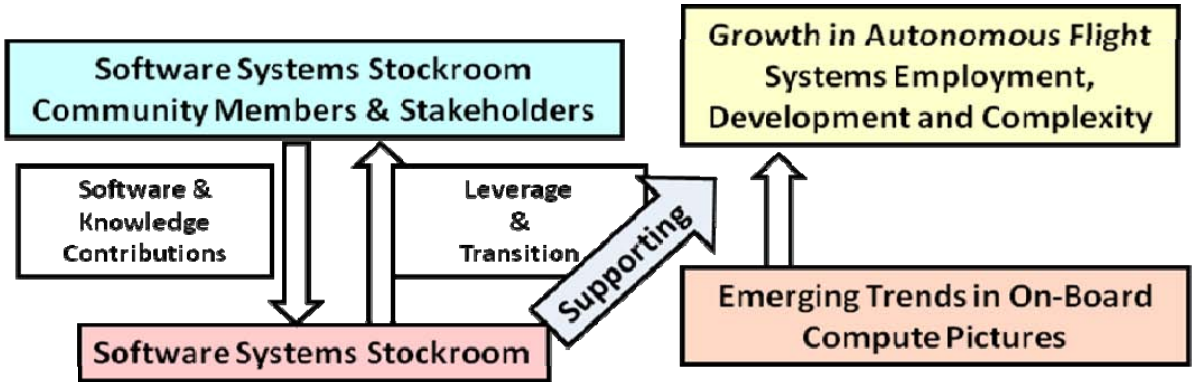


Figure 1: Relationships of S3 Components

Community members and stakeholders are shown in more detail in Figure 2, many of whom have provided guidance and assessment during the development of this S3 prototype and its domain contents. DoD acquisition programs will benefit from leveraging this domain content in terms of lower software development costs. The DoD contractor community, both large and small businesses, will also benefit from more affordable systems development. Academic researchers will be able to post their tools, algorithms, and software on the S3. The defense community will be able to post domain specific information about their software and tools for leverage by systems developers and other government researchers.

Figure 2: S3 Community Members and Stakeholders

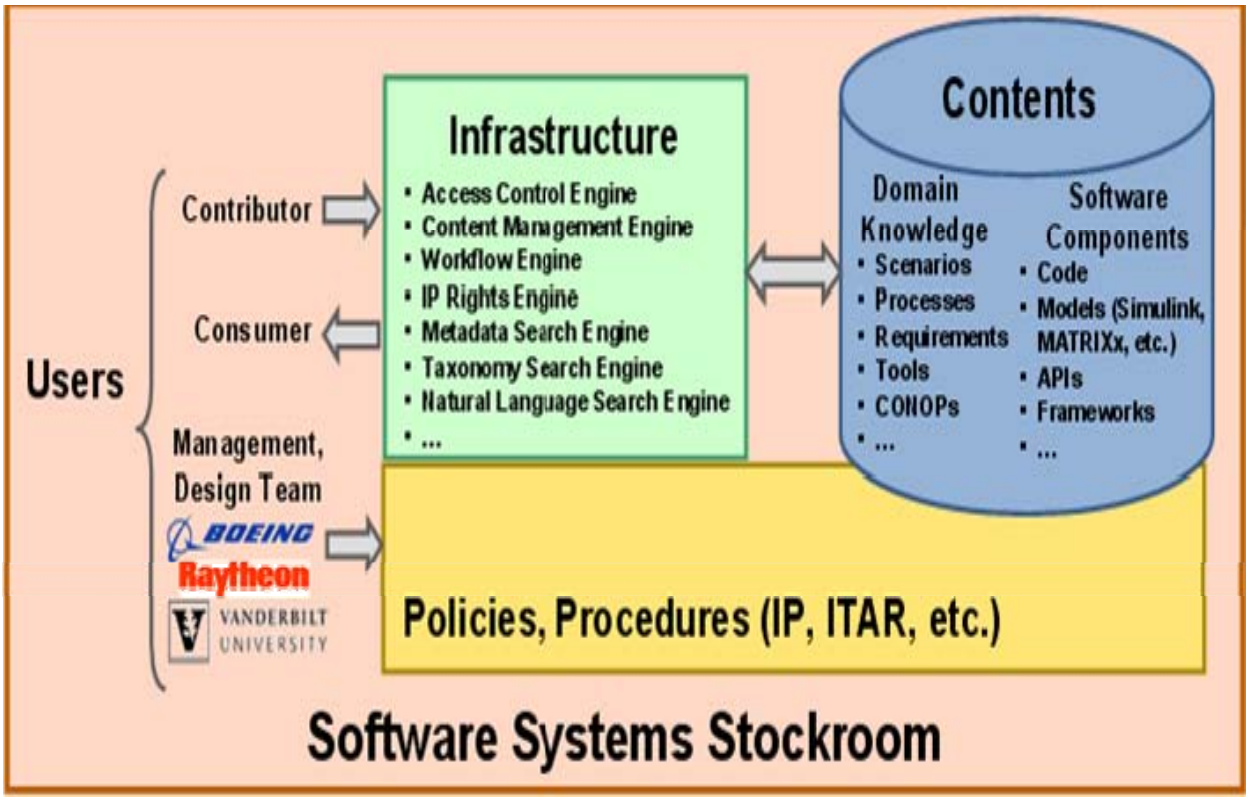


Figure 3: S3 Infrastructure Capabilities and Repository Contents

In Figure 3 is presented the infrastructure capabilities and a listing of envisioned types of the repository contents of S3.

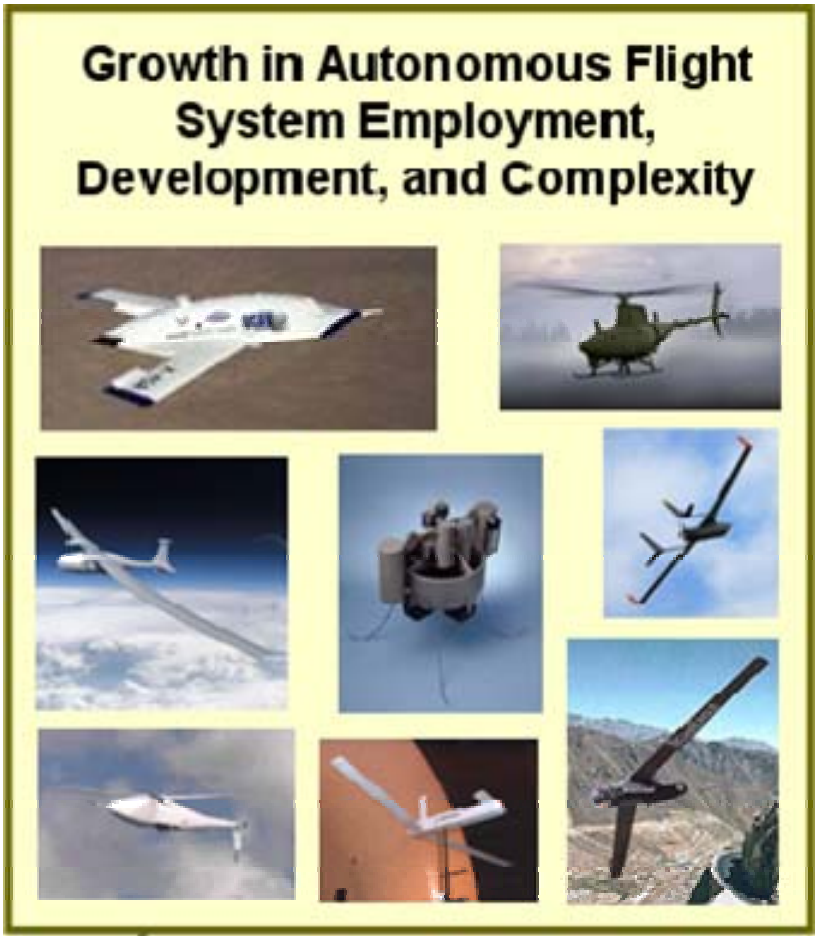


Figure 4: Evolving Autonomous Flight Systems

The domain of autonomous flight systems (Figure 4) includes UAVs and smart weapons, and was chosen as the domain to support for a number of reasons, including that UAVs and smart weapons represent a DoD growth area within all the armed services.

The aspect of the supported domain associated with hosting embedded flight software on emerging processing systems (e.g., multi-core) is also significant (Figure 5). Multi-core architectures will soon become the de-facto embedded processor architecture available to the mil-aero industry, and work needs to be done to allow for safe and secure hosting of flight software on these embedded compute platforms. With the desire for increased on-board capabilities on existing and smaller aircraft, this element of the S3 domain can provide important support for those needing help solving multi-core development issues.

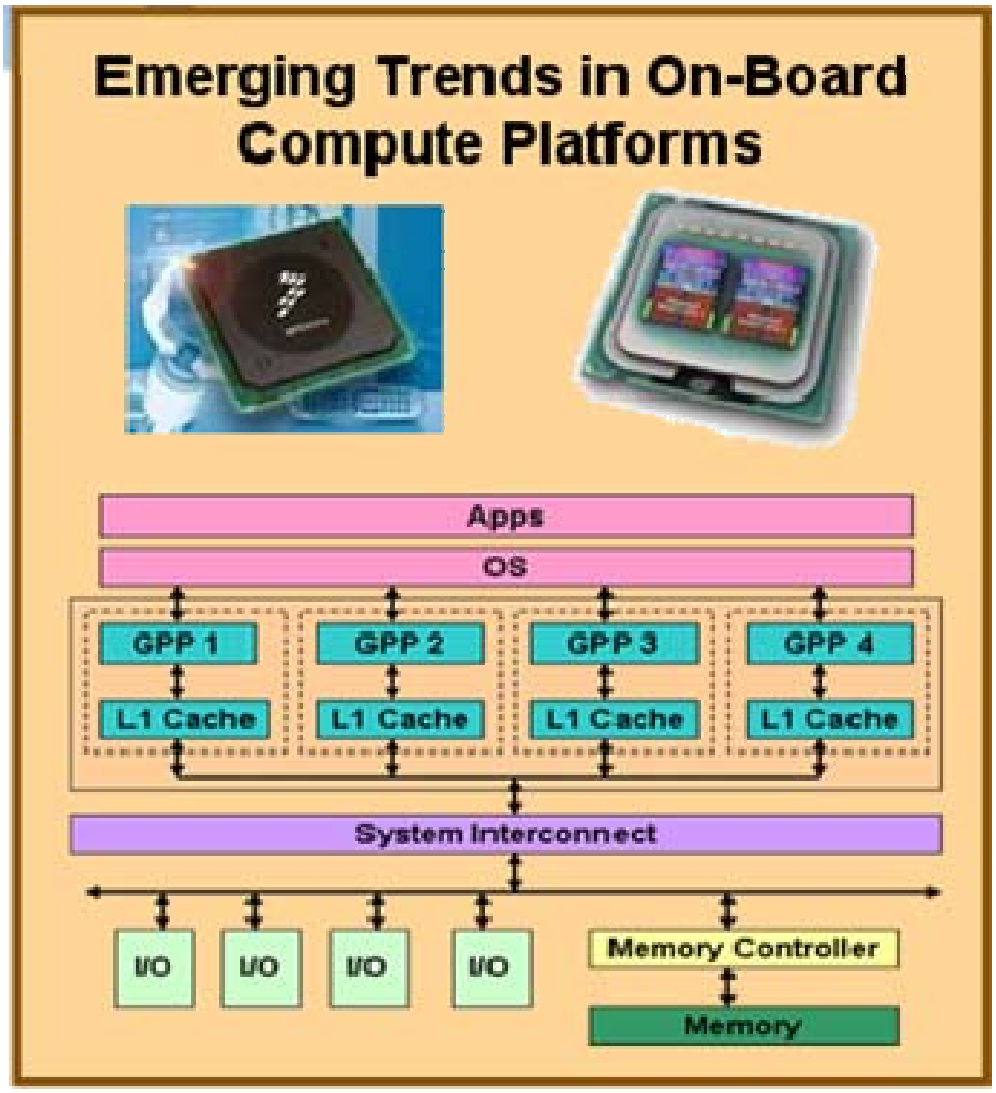


Figure 5: Emerging On Board Compute for Airborne Platforms

On-board compute platforms are becoming more pervasive in warfighting, in part because they support military objectives while keeping friendly forces out of harm's way. The S3 goal of supporting a mix of on-board vehicle management and mission management, including functions that were performed by aircrew on manned platforms, provides significant software producibility challenges for autonomous flight systems (Figure 6). The industry component of our team, Boeing and Raytheon, has great interest in this class of compute platforms, as do our mil-aero industry competitors. And, in terms of building a wider community, there is significant interest in this area from other industries (embedded software providers, tools developers, etc.), small business, and academia.

Figure 6: New and Future Autonomous Flight Systems

2.3. Scope

This report details the Phase 1 activities involved in the development of the prototype S3, in populating the prototype with an initial set of artifact contents, and in engaging a community of domain knowledge suppliers and users working this domain. Also included in this report is a description of the as-built capabilities that have been implemented in the S3 prototype and features of S3 contents that have been populated in Phase 1.

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

The program overview, assumptions, execution summary, methods, and procedures are presented in this section.

3.1. Program Overview and Assumptions

With the S3 program, AFRL sought to "improve the capability to produce software for DoD systems by encouraging the capture and reuse of domain knowledge and expertise through an open, community-driven, technically focused shared infrastructure." [1]

In support of this vision, Boeing, Raytheon, and Vanderbilt University put together a team to build and populate an S3 that supported the development of autonomous flight systems embedded software on emerging processing systems. Assumptions associated with this approach included the following:

- The autonomous flight systems domain was of such importance (currently a DoD growth area for all the armed services, software complexity is resulting in producibility issues) that investment in this domain would be deemed beneficial.
- A concentration on support for the development of embedded flight software on emerging processing systems (e.g., multi-core) is also of great value.
- An S3 development team that includes more than one mil-aero contractor (Boeing and Raytheon), which are often competitors, would result in an open solution not dominated by a single contractor.
- An S3 development team that brought in a university teammate (Vanderbilt University) would result in an S3 that leverages the most recent developments in web and content management technologies, and provide a contractor-neutral organization for hosting S3 (making S3 more inviting to other contractors).

The team organization for this effort is shown in Figure 7. Boeing led program execution, led development of autonomous flight system-related aspects of program, and populated key domain artifact elements into S3. Raytheon led the Use Case development and the multi-core-related elements of the effort, and also populated major domain artifact elements into S3. Vanderbilt University led the S3 prototype development and functioned as S3 administrators after the S3 was made available on the Internet in Mar 2009. Various consultants from our program team in the areas of vehicle control, multi-core technology, and software reuse were called upon during program execution, but the majority of the work was accomplished by the PIs along with Doug Stuart and Chris van Buskirk.

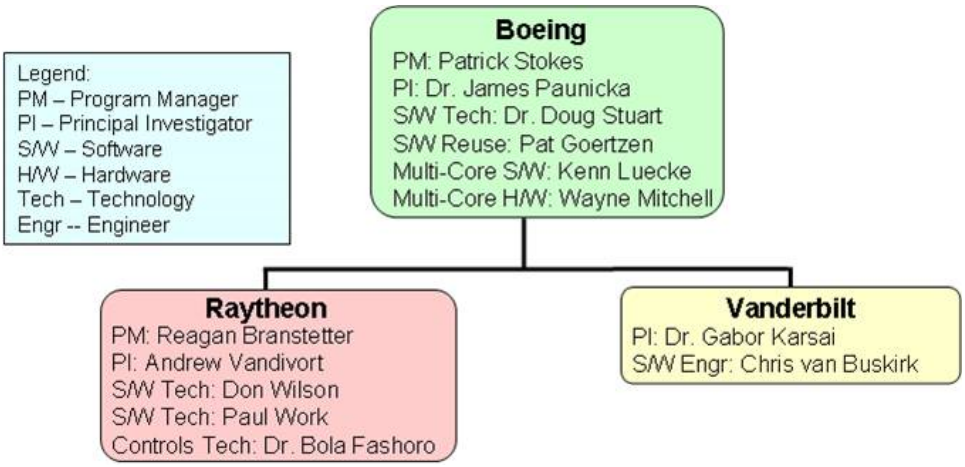


Figure 7: Boeing S3 Team Organization

3.2. Execution Summary, Methods, and Procedures

Program period of performance was Feb to Aug 2009. Major events occurring during program execution are summarized in Table 1.

Table 1. Significant Program Events

Event	Date(s)	Location	Organizations / People Involved	Synopsis
PoP Start	05 Feb 2009		Boeing	
Kickoff Meeting	26 Feb 2009	Arlington, VA	OSD, AFRL, ARL, NRL, ONR, Boeing, Raytheon, Vanderbilt	Boeing-led team briefed program plans for prototype development and community engagement
Software and Systems Producibility Collaboration and Experimentation Environment (SPRUCE) Technical Interchange Meeting	19 Mar 2009	Telecon / Webex	Boeing, Raytheon, Vanderbilt, Lockheed SPRUCE Team	Discussed what elements of SPRUCE that can be leveraged by S3 and preliminary exploration of how to implement that leverage in software

Event	Date(s)	Location	Organizations / People Involved	Synopsis
S3 Stood up on Internet with https Secure Socket Layer protection	31 Mar 2009	Vanderbilt servers in Nashville, TN	Boeing S3 team, User Community	S3 accessible on internet with username and password; used by multiple members of the user community to share domain knowledge in files; also used by the S3 team to share S3 design information
AIAA InfoTech @ Aerospace Conference	09 Apr 2009	Seattle, WA	Boeing,	Boeing briefed S3 in a presentation entitled, "An OSD Software Systems Stockroom Instance Supporting Autonomous Flight Systems," to a national audience including industry, academia, and government
IEEE Real-Time and Embedded Technology and Applications Symposium	16 Apr 2009	San Francisco, CA	Boeing	Boeing briefed S3 to a national audience including industry, academia, and government
Midterm Review	05 May 2009	Arlington, VA	AFRL, ARL, NRL, ONR, Boeing, Raytheon, Vanderbilt	Boeing-led team briefed current design of the S3 prototype & current community engagement status, performed a demonstration of the S3 prototype, and briefed plans for further prototype development and community engagement
Safe & Secure Systems & Software Symposium	02 Jun 2009	Dayton, OH	AFRL, Boeing, Vanderbilt	Boeing briefed S3 to a national audience including industry, academia, and government
Final Review	15 Jul 2009	Boeing Office, Arlington, VA	AFRL, ARL, NRL, ONR, Boeing, Raytheon, Vanderbilt	Boeing-led team briefed final design of the S3 prototype & domain knowledge contents, and Phase 2 plans; and performed a demonstration of the S3 prototype
PoP End	01 Aug 2009		Boeing	

Throughout the execution of the program, team coordination was enabled with the use of a number of methods, procedures, and tools. Weekly design and coordination sessions were held over teleconferences and Webex that involved Boeing personnel in St. Louis, Missouri; Raytheon personnel in Tucson, Arizona; and Vanderbilt University personnel in Nashville, Tennessee. These weekly sessions were augmented with additional virtual meetings when necessary. Sharing of data products (design information, reports, presentations, etc.) was facilitated with an S3 SharePoint program data site hosted at Boeing, and later with a password-protected file share on the S3 itself after it was stood up on the Internet in Mar 2009. Teleconferences and Webex were also used for discussions with the Software and Systems Producibility Collaboration and Experimentation Environment (SPRUCE) development and operations team on possible leverage of their infrastructure by S3.

For major program events, such as the Kickoff, Mid-Term, and Final Reviews, we would convene for face-to-face sessions at the Boeing Washington DC Office (WDCO) to finalize presentation material.

For collaboration with our AFRL customer, we participated in bi-weekly teleconferences to communicate technical status and plan future program events. These calls were open to, and attended by, our extended team members from Raytheon and Vanderbilt University. During these calls, our team would communicate status, raise questions and issues, and seek feedback from our customers. AFRL would communicate important program status and plans, make us aware of any programmatic issues, and comment on our current approach to S3.

A Basic Support for Cooperative Work (BSCW) Internet-based shared workspace site (https://bscw.sei.cmu.edu/bscw/bscw.cgi) was used for exchanging data generated by the multiple participants who attended government-hosted S3 meetings.

The AFRL Jiffy program management system (https://jiffy.rl.af.mil) was used by Boeing to deliver Contract Data Requirements List (CDRL) documentation, including periodic status reports and other technical reports, over the public internet.

The development of the prototype was driven, in part, by early definition of User Roles and Use Cases, led by Raytheon with contributions from Boeing. At the same time, Vanderbilt began S3 prototype development leveraging the native capabilities of the Drupal Content Management System [2], augmented with custom tailoring to match the evolving Use Cases. To support domain-specific repository features, Boeing led the development of domain-specific ontologies for S3, with contributions from Raytheon. This ontological information was integrated into S3 with tools configured by Vanderbilt.

Testing of the evolving prototype, whose home page is shown in Figure 8, was accomplished with a semi-formal process involving initial tests by Vanderbilt, followed by additional exercising of new functionality by Boeing and Raytheon.

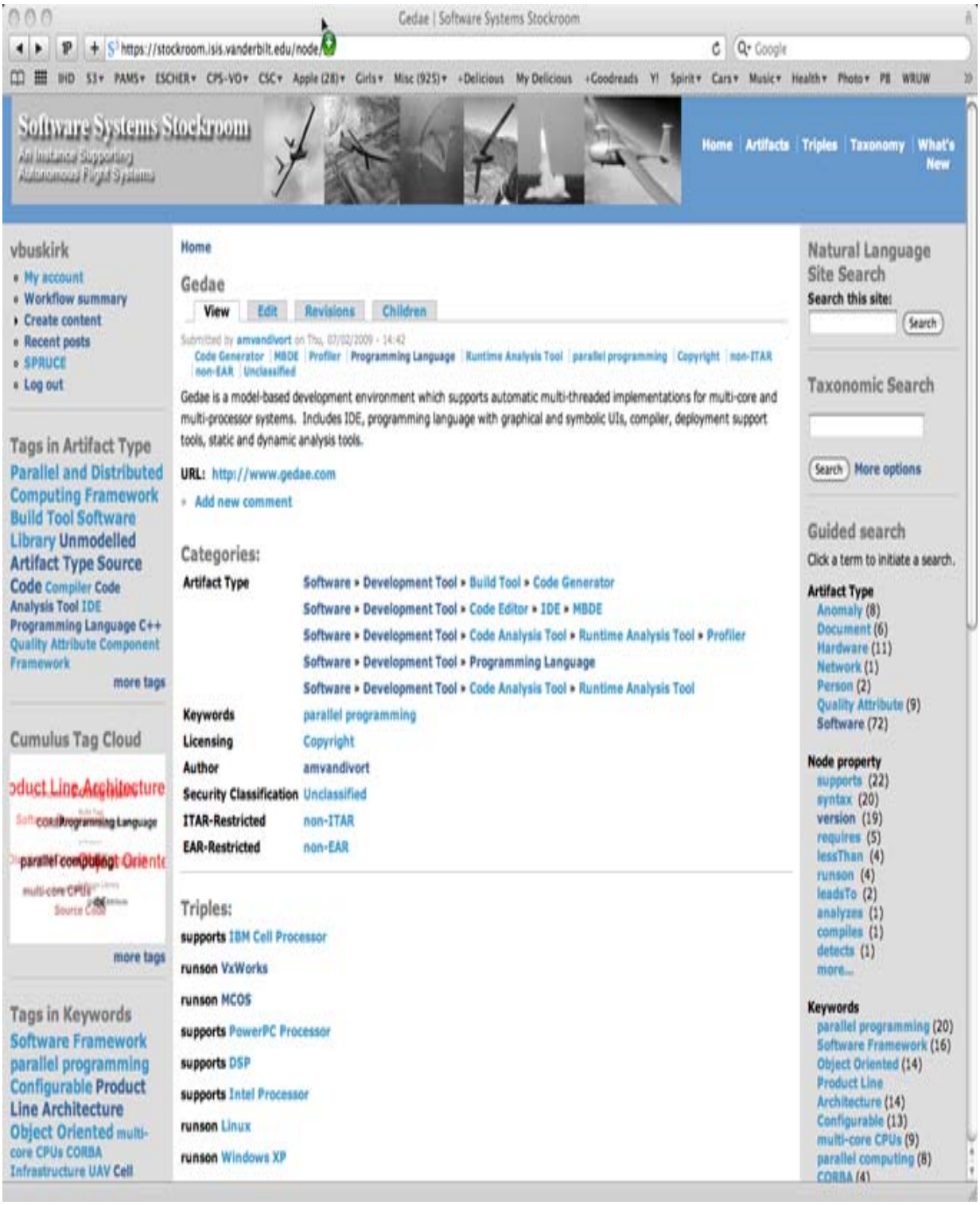


Figure 8: S3 Prototype Home Page

4.0 RESULTS AND DISCUSSION

The following subsections provide details on the results generated during the program. The discussion begins with the results of our Use Case development efforts. This is followed by details of the S3 prototype as built on the program during Phase 1, including a summary of S3 capabilities, the resulting S3 architecture, and a summary of S3 domain contents. This is followed by a discussion of community activities on the program.

4.1. Users and Use Cases

The Stockroom prototype identifies 14 user roles (actors) and 33 system-level use cases, which are further organized in to three categories: Artifact-centered, Collaboration, and Administration.

4.1.1. Users

As shown in Figure 9, the Stockroom has three primary user roles: Contributor, Consumer and S3 Administrator.

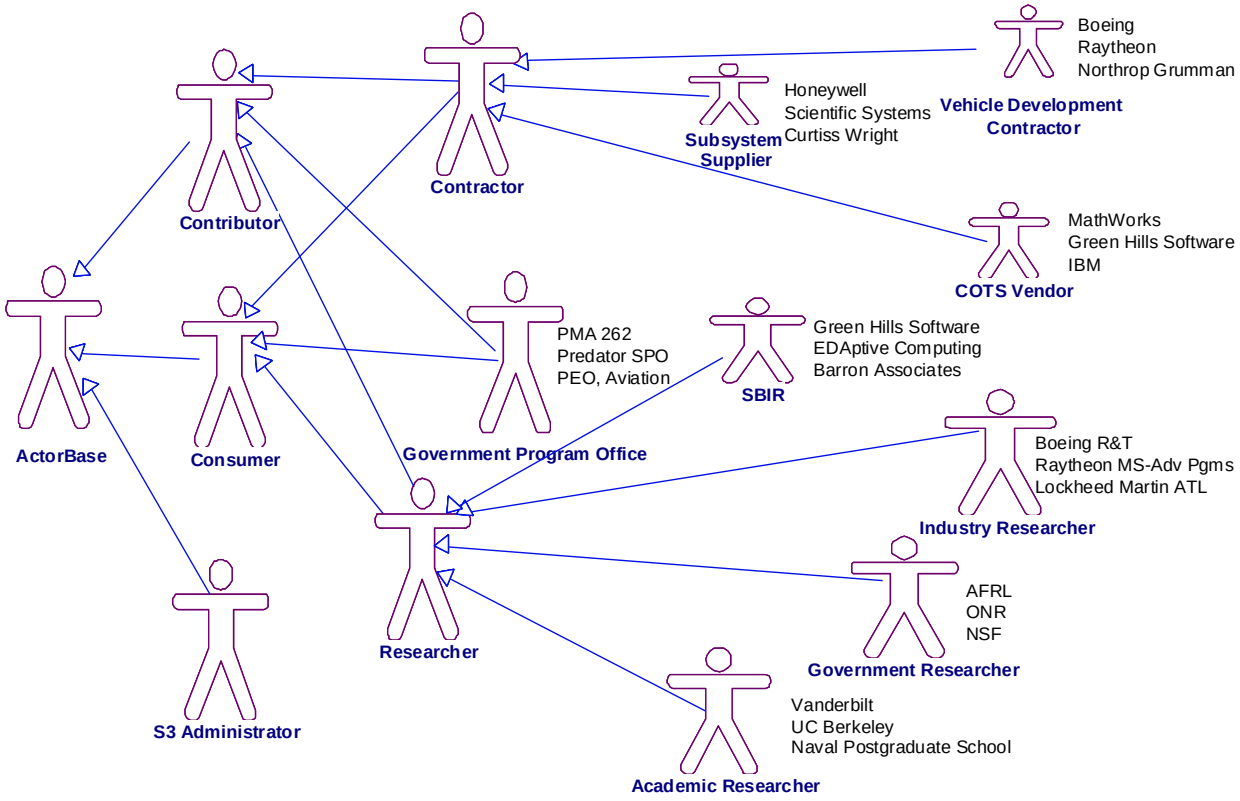


Figure 9: Stockroom Roles and Potential Users

Ten additional user roles are identified which are refinements of the primary roles. (These refined roles primarily serve as taxonomy of user types for the Stockroom's role-based access control engine.) A top level user role (ActorBase) is also provided to capture functionality common to all users. A brief description of each user role is provided below:

User Descriptions:

ActorBase – base user role (all other user roles inherit from ActorBase). This role primarily serves to capture user metadata common to all users.

Contributor – contributor role submits content in the form of artifact(s) to the Stockroom, which may include associated metadata, e.g., related keywords, artifact type (taxonomic classification), and Resource Description Framework (RDF) triples (ontological relationships).

Consumer – content user role, which locates (via natural language, metadata, or ontology-based search), acquires (e.g., downloads) Stockroom content; and may also use and provide feedback on content artifacts.

S3 Administrator – Stockroom administration role, responsible for maintenance and administration of users, working groups, content and metadata. Also maintains the taxonomic and ontological repository data and supervises the evolution of these data structures. Responsible for maintenance of, and enhancements to, Stockroom hardware and software infrastructure.

Contractor – inherits from both the Contributor and Consumer user roles; provides a unique user role for the (industrial) contracting community.

Subsystem Supplier – inherits from the Contractor user role, providing a further refined role for subsystem suppliers (e.g., avionics, flight control sensors, algorithm suppliers, etc.).

COTS Vendor – inherits from the Contractor role, providing a further refined role for members from the commercial-off-the-shelf vendor community. Examples would include vendors of commercial software and hardware products (e.g., software tools, real-time operating systems (RTOSs), embedded processor products, etc.).

Vehicle Development Contractor – inherits from the Contractor role, providing a further refined role for contractors from the UAV and intelligent airborne weapon (e.g., missile) community.

Government Program Office - inherits from both the Contributor and Consumer user roles; provides a unique user role for government program office representatives.

Researcher – inherits from both the Contributor and Consumer user roles; provides a unique user role for members of the research community.

SBIR – inherits from the Researcher user role, providing a further refined role for small business members participating in a Stockroom-related SBIR or Small Business Technology Transfer (STTR) program.

Industry Researcher – inherits from the Researcher role; provides a further refined researcher role for industrial researchers (e.g., from the Mil-Aero community) conducting or participating in Stockroom-related research activities.

Government Researcher – inherits from the Researcher role; provides a further refined researcher role for government researchers (e.g., AFRL, ONR, ARL, National Science Foundation (NSF), etc.) conducting or participating in Stockroom-related research activities.

Academic Researcher – inherits from the Researcher role; provides a further refined researcher role for academic researchers conducting or participating in Stockroom-related research activities.

4.1.2. Use Cases

Artifact centered, collaboration, administration use cases are presented in this section.

4.1.2.1. Artifact Centered Use Cases

The Unified Modeling Language (UML) use case diagram shown in Figure 10 identifies the 11 Artifact Centered use cases and their relationships to one another.

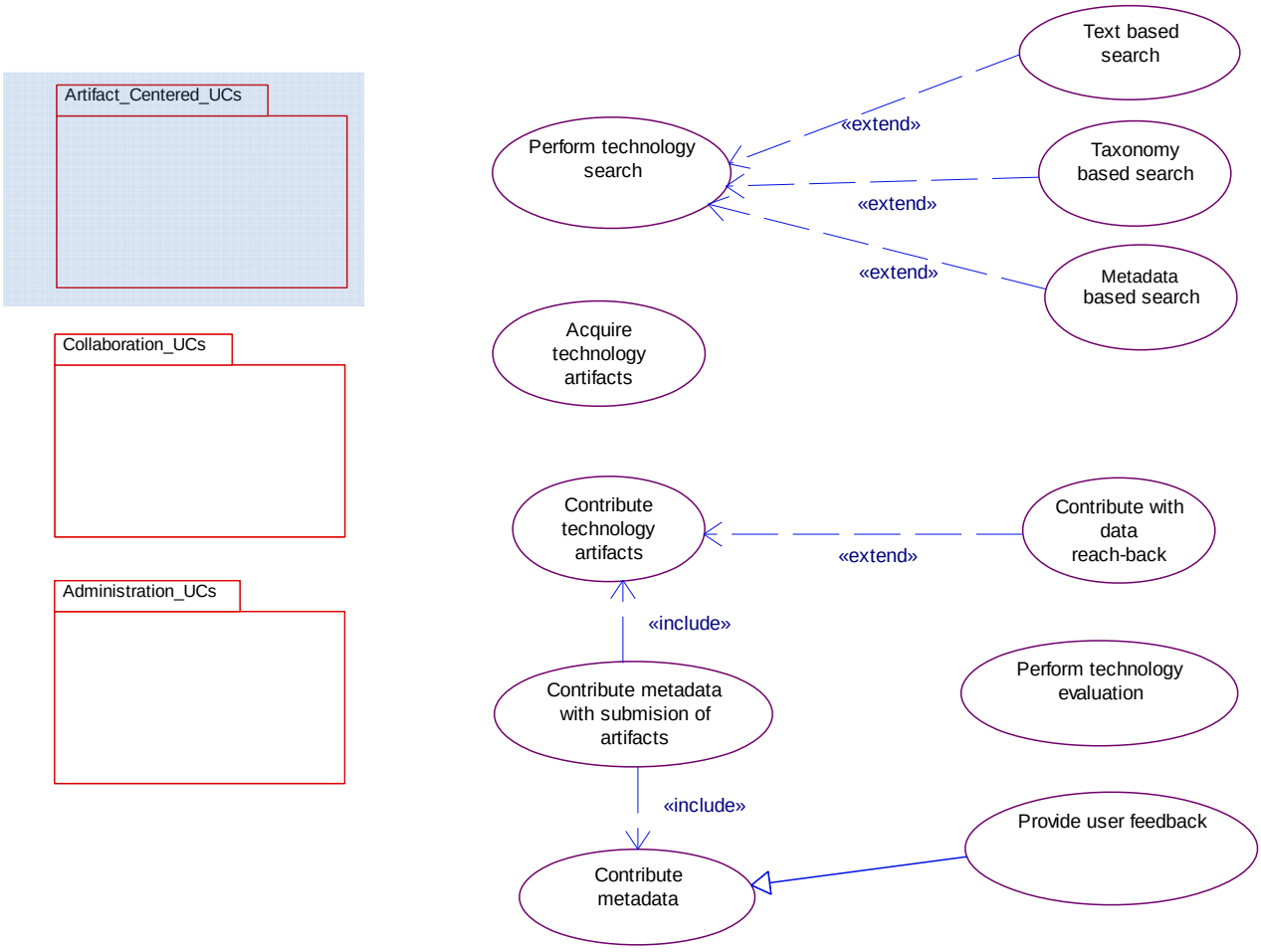


Figure 10: Artifact-centered Use Cases

Descriptions of each use case are provided below:

Artifact Centered Use Case Descriptions:

Perform technology search – use case providing for the searching of Stockroom artifacts. This use case optionally provides three alternative search capabilities: Text based search, Metadata based search, and Taxonomy based search. (Each of these search alternatives are described below.) The Stockroom user invokes one of the alternative search engines via the Stockroom user interface (UI), which is a standard web interface with dialog boxes for each of the various forms of search. Search results are returned to the user's browser. Search results consist of a brief summary and Uniform Resource Locator (URL) for each resultant.

Text based search – use case which performs a text based (i.e., natural language) search for specified words and/or phrases. Search targets will include all text associated with Stockroom artifacts, and will also include Technical Working Group (TWG) and Community Standards Working Group (CSWG) forum and blog postings.

Metadata based search – use case which performs a search of Stockroom artifacts for specified metadata words or phrase. Search targets include metadata types, including keywords.

Taxonomy based search – use case which performs taxonomy and ontology based searches. Taxonomy based search is supported via the Guided Search facility, found on the Stockroom homepage. This facility supports "drill-down" style search of the taxonomy hierarchy, narrowing the scope of search in a step-wise manner. Full ontology-based search is currently supported via an exported Web Ontology Language (OWL) [3] file (s3.owl) and a COTS OWL reasoner (Protégé [4] FACT reasoning engine [5]). (The intention in Phase 2 is to fully integrate the ontological search capability into the Stockroom web portal.)

Acquire technology artifacts – use case providing for the downloading of an artifact data payload. Two scenarios are supported: 1) direct download of data file(s) from Stockroom via artifact web page; 2) acquisition of data (or material) from 3rd party (e.g., intellectual property (IP) owner, COTS vendor, etc.) via URL reference on artifact web page.

Contribute technology artifacts – use case providing for the contribution of technology artifact to Stockroom. This use case consists of two basic steps: 1) creation of repository artifact, using the Stockroom's *Create Content* facility; 2) upload of artifact data payload using the *File attachment* utility (this step is optional; artifacts are not required to have data payloads).

Contribute with data reach-back – use case which optionally extends Step 2 of the Contribute technology artifacts' use case to include a URL link to a 3rd party web-site instead of (or in addition to) local data payload. This use case allows for artifact creation (and associated metadata) without requiring collocation of artifact data payload (e.g., for purposes of IP protection, security, International Traffic in Arms Regulations (ITAR) / Export Administration Regulations (EAR) compliance, etc.).

Contribute metadata – use case which provides for the addition of relevant metadata to be added to a Stockroom artifact. At the creation of an artifact, a minimal metadata set is required to be supplied by the artifact author; however, it is also anticipated that additional metadata will be supplied at other times by the artifact author and other users with sufficient user role privileges. This could include all metadata types, but the most commonly anticipated scenario would be the addition of relevant RDF-triples, in which the artifact in question would serve as either the subject or object of such a relationship. This is one of the user-based mechanisms of Stockroom ontology evolution.

Provide user feedback – a refinement of the Contribute metadata use case, which specifically provides for user feedback with regards to a Stockroom artifact. Each artifact has a comments metadata field, which allows user roles with sufficient privilege to enter wiki-style comments. User comments appear on the artifact web page.

Contribute metadata with submission of artifacts – use case which includes both Contribute technology artifacts and Contribute metadata use cases. In this use case an artifact is created as prescribed in the former, and is simultaneously populated with additional metadata (e.g., RDF-triples) as described in the latter, serving to "anchor" the artifact into the Stockroom ontology.

Perform technology evaluation – use case which involves the (typically collaborative) evaluation of one or more Stockroom artifacts. This would typically be a collaborative activity performed under the auspices of a Stockroom TWG. This use case envisions the cross-leverage of the SPRUCE program experimental resources – to the extent this is possible.

4.1.2.2. Collaboration Use Cases

The UML use case diagram shown in Figure 11 identifies the 14 Collaboration Centered use cases and their relationships to one another.

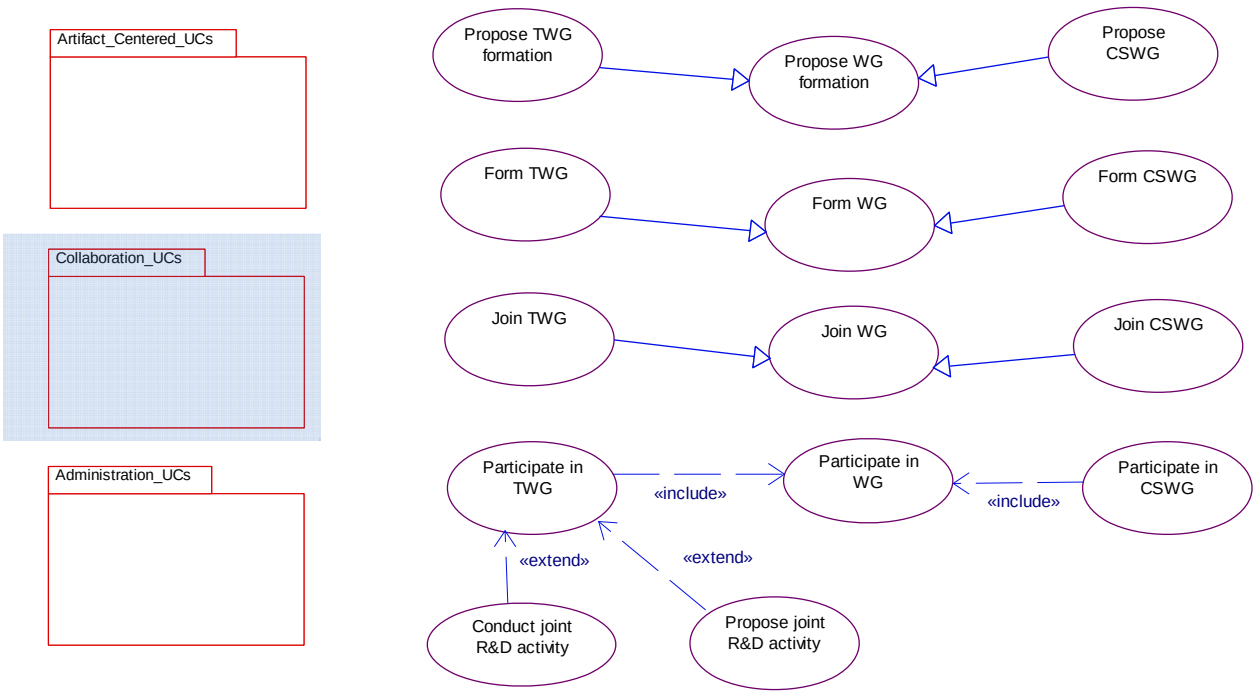


Figure 11: Collaboration Use Cases

Descriptions of each use case are provided below:

Collaboration Use Case Descriptions:

Propose WG formation – use case for the formation of a Stockroom working group. It is envisioned that WG formation would be requested by at least two interested parties. If approved by Stockroom governance, a request for workgroup formation would be forwarded to the Stockroom Administrator.

> *Two forms of Stockroom working groups are provided for: TWGs and CSWGs. It is believed TWGs will be the more common of the two species, and that CSWGs will be an outgrowth of at least some TWGs – where drafting of standards becomes a community objective.*

Propose TWG formation – refinement of Propose WG formation use case in support of a TWG.

Propose CSWG formation – refinement of Propose WG formation use case in support of a CSWG. Similar to parent use case, except it is envisioned that CSWG formation would be requested by an existing TWG chair, and would be an outgrowth of the TWG the chair represents.

Form WG – use case initiated by Propose WG formation use case. Upon receipt of new request Stockroom Administrator would set-up required forum and web file manger structures for new working group. Administrator would also set appropriate role-based access privileges for members of new working group.

Form TWG – refinement of Form WG use case in support of a TWG.

Form CSWG – refinement of Form WG use case in support of a CSWG. Formation of CSWG would require concurrence of Stockroom governance body.

Join WG – use case which addresses request of a Stockroom user to be added to existing working group. Request (currently in the form of an email) would be submitted to working group chair. If approved, the chair would submit a request to Stockroom Administrator to add user to working group. Administrator would then set appropriate role-based access privileges for the new working group member. Concurrence of Stockroom governance may be required for some working groups.

Join TWG – refinement of Join WG use case in support of a TWG.

Join CSWG – refinement of Join WG use case in support of a CSWG.

Participate in WG – use case addressing activities associated with participating in a working group. In the Phase 1 Stockroom prototype two main collaboration mechanisms are provided: 1) Web-based file manager, for sharing of data files amongst the members of a working group; 2) Web-base forum communication utility, which allows for topic separated, bulletin board style communications amongst forum members.

> *It is anticipated activity patterns (e.g., policies, procedures, methodologies, etc.) associated with a specific working group will be highly specific to the objectives of that community.*

Participate in TWG – refinement of Participate in WG use case in support of a TWG.

Participate in CSWG – refinement of Participate in WG use case in support of a CSWG.

Propose joint R&D activity – use case supporting proposal of joint Research and Development (R&D) activity. It is envisioned that proposal will be requested by one or more TWG chairs to the Stockroom Administrator, and would be conducted under the auspices of same TWGs.

Conduct joint R&D activity – use case which supports the conduct of joint R&D activity. Upon receiving joint R&D proposal, the Stockroom Administrator will submit proposal to Stockroom governance body. If approved, Administrator will work with TWGs to configure necessary artifacts and infrastructure to support and conduct joint R&D activity. As in Perform technology evaluation use case, cross-leverage of the SPRUCE program experimental resources will be explored where possible.

4.1.2.3. Administration Use Cases

The UML use case diagram shown in Figure 12 identifies the 8 Administration use cases and their relationships to one another.

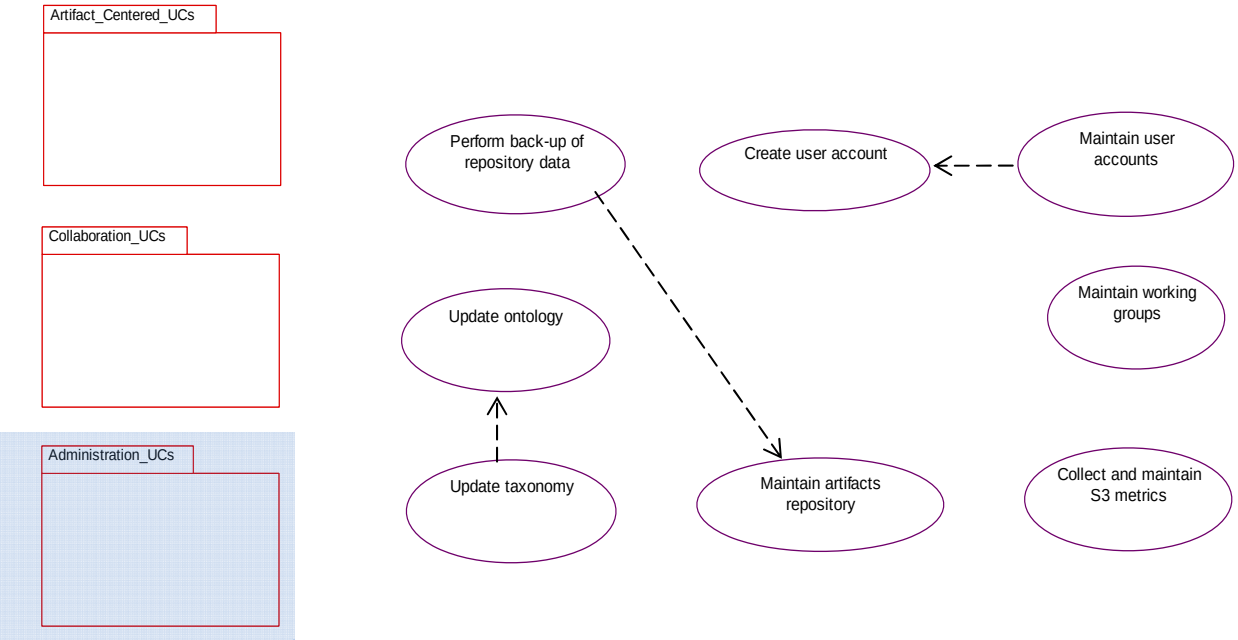


Figure 12: Administration Use Cases

Descriptions of each use case are provided below:

Administration Use Case Descriptions:

Perform back-up of repository data – use case providing for the administration and support of daily Stockroom repository data back-up (and restore, as required).

Create user account – use case providing for the generation and configuration of Stockroom user accounts.

> *The Phase 1 Stockroom prototype design intended to leverage the SPRUCE user authentication service. Due to time and funding constraints this was not feasible for the prototype, but will continue to be pursued during the Phase 2 implementation.*

Maintain user accounts – use case providing for the maintenance of user accounts, including: decommissioning of old accounts, adjustment and reconfiguration of role-based access privileges, etc.

Maintain working groups – use case providing for the maintenance of Stockroom working groups (both TWGs and CSWGs).

Maintain artifacts repository – use case providing for the maintenance of Stockroom artifacts repository. E.g., managing the addition of processing and data storage, performing routine data integrity checks, etc.

Update ontology – use case providing for the update and evolution of the Stockroom ontology. This might include activities such as routinely examining "emerging" user supplied ontology extensions (e.g., user generated RDF-triples) and keywords for potential modification to "planned" ontology kernel. These decisions would be made by the Stockroom Ontology Control Board CSWG, and would be implemented by the Stockroom Administrator.

Update taxonomy – use case providing for the update and evolution of the Stockroom taxonomy. This might include activities such as routinely examining "emerging" user supplied ontology extensions (e.g., user generated RDF-triples) and keywords for potential modification to the taxonomy type hierarchy. These decisions would be made by the Ontology Evolution Steering Committee CSWG (Section 4.2.2.2.1.2), and would be implemented by the Stockroom Administrator.

Collect and maintain S3 metrics – use case providing for the collection and dissemination of Stockroom usage metrics, including: number of users, number of nodes, number of repository artifacts, number of web files, number of site visits (including pages, hits and bandwidth usage).

4.1.2.4. Example Scenarios

Figure 13 shows a UML sequence diagram of the Create User Account use case.

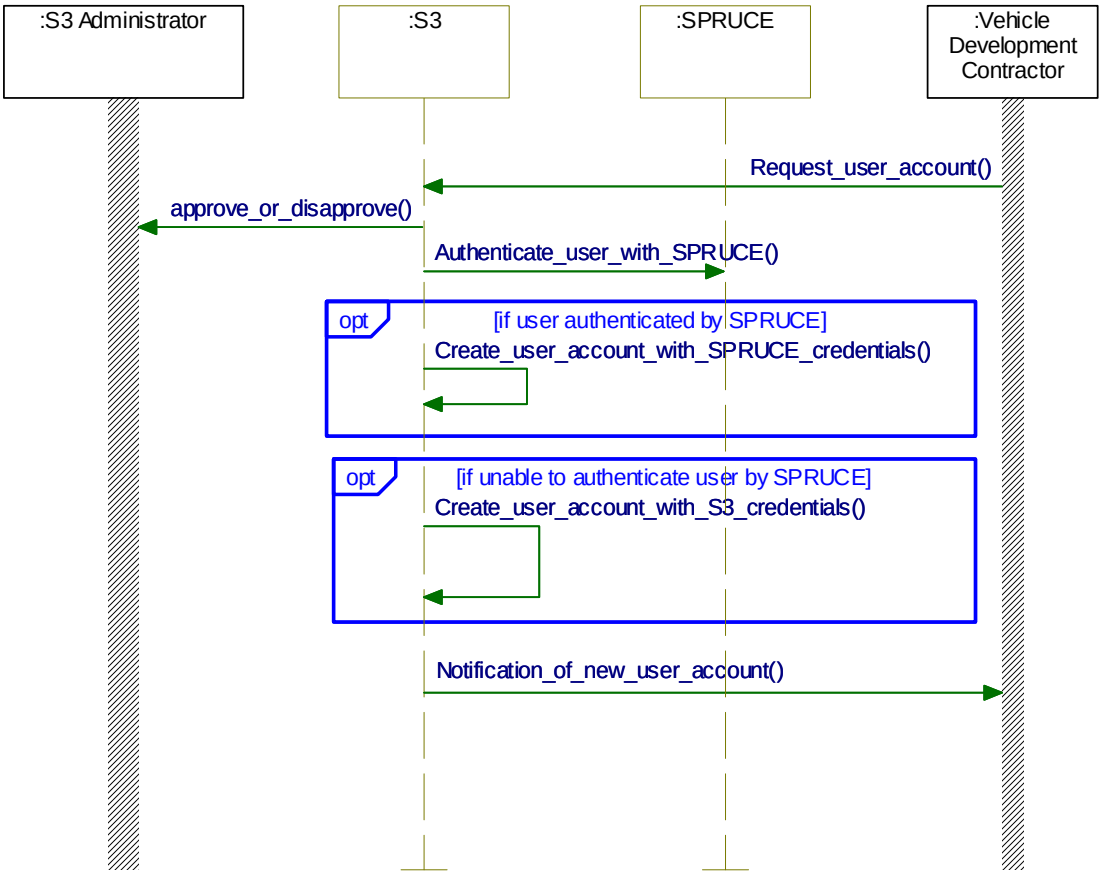


Figure 13: Create User Account Use Case

In the particular example shown in the figure, a member of the Vehicle Development Contractor user role is submitting a request for a user account to the Stockroom web portal. The request is forwarded to the S3 Administrator for approval. In this diagram two alternative sequences of events are shown. In the first, an attempt is made to authenticate the user via the SPRUCE web portal. If this were to result in success, the Stockroom user account would be created and the Vehicle Development Contractor would be notified. Alternatively, if the user could not be authenticated through the SPRUCE web portal, then a new account would be created with S3 credentials, and the new user would be similarly notified.

In Figure 14 we show an example scenario involving three use cases. At the top of the figure we see the S3 Administrator initiate the Form TWG use case and create a new UAV/UAS Autonomous Flight Systems TWG. Next, the Stockroom is seeded with domain artifacts and related metadata in support of the new UAV/Unmanned Aircraft Systems (UAS) TWG (exercising the Contribute Metadata with Submission of Artifacts use case). Finally, we have a member of the Vehicle Development Contractor user role, seeking middleware technologies suitable for a UAV flight software application, who becomes aware of the UAV/UAS TWG and requests to join the TWG, exercising the Join TWG use case. (UAV/UAS TWG membership is granted in this example scenario.)

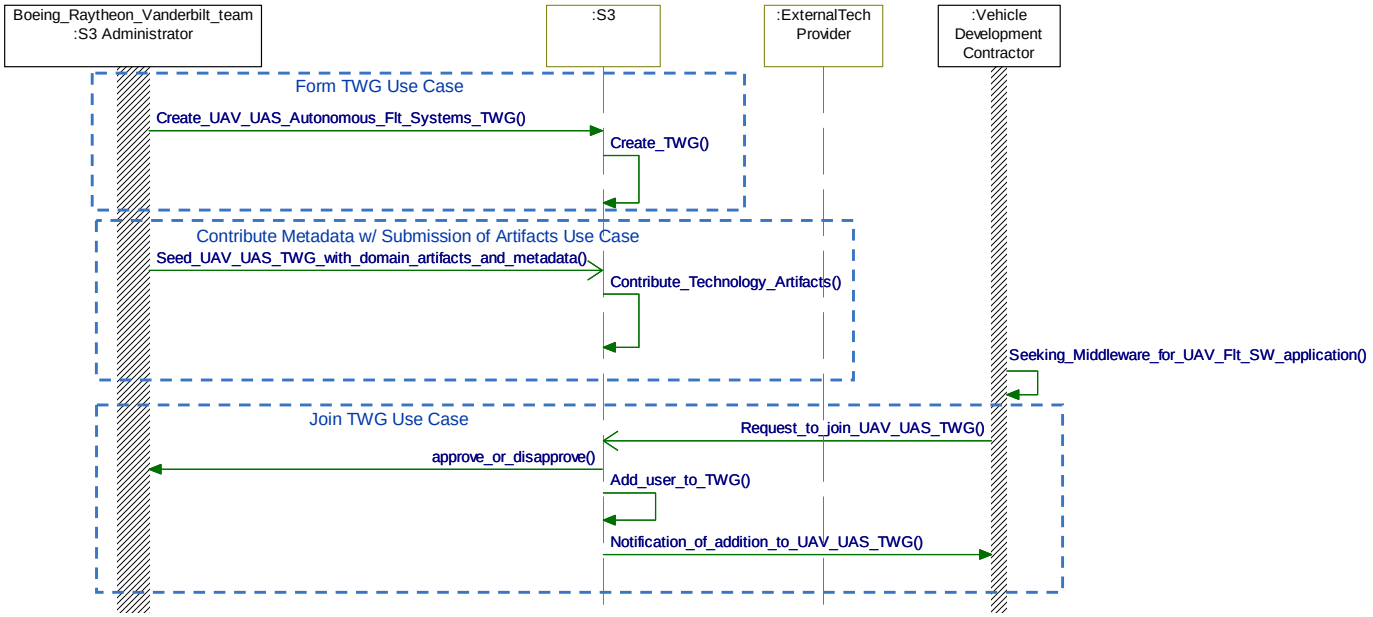


Figure 14: Technical Working Group (TWG) Example

Figure 15 demonstrates the Perform Technology Search use case followed by the Acquire Technology Artifacts use case. Here we see the Vehicle Development Contractor sequentially perform text-based, taxonomy-based and metadata-based searches for middleware technology solutions. The Vehicle Development Contractor then evaluates the returned search results, and ultimately identifies a Stockroom artifact she wishes to download. The Vehicle Development Contractor then proceeds to access the Stockroom artifact (by clicking on the artifact URL). From the artifact web page the artifact payload is downloaded from the Stockroom database repository if the artifact payload is hosted at the Stockroom. Alternatively, if the artifact payload is not located at the Stockroom, the payload may be acquired from a remote site provided by an external technology provider (data reach back example).

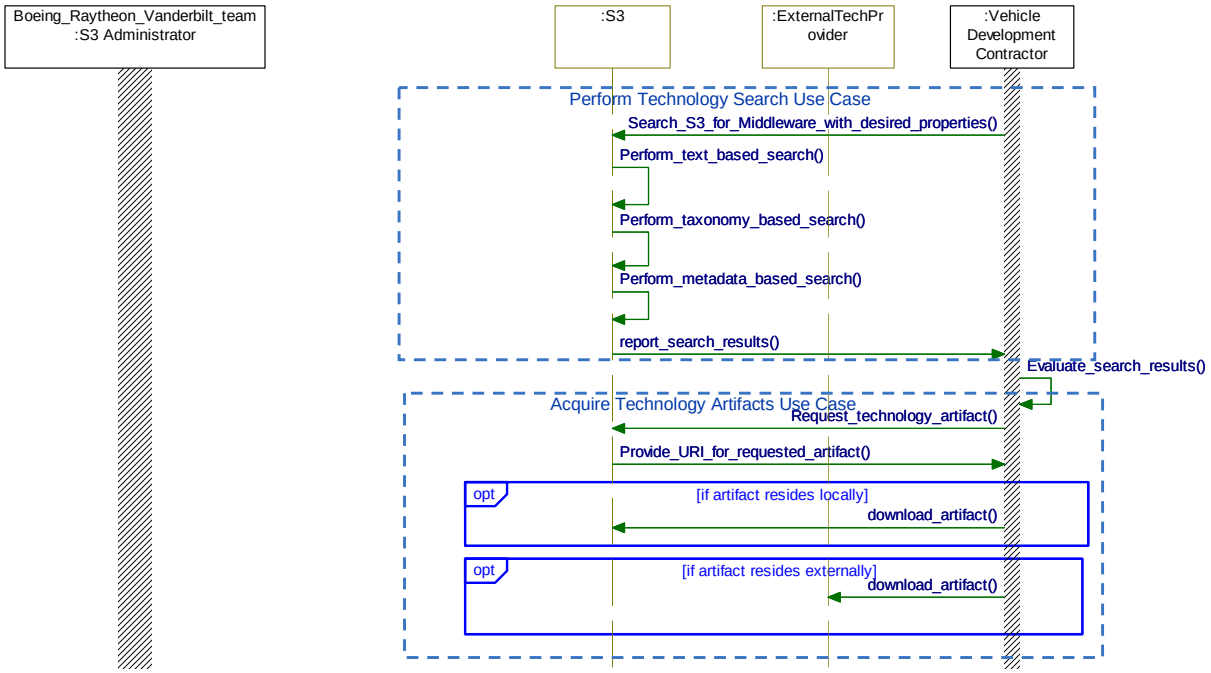


Figure 15: Example of Technology Search and Artifact Acquisition

Figure 17 are graphical depictions of two related scenarios involving the search and acquisition of technology solutions via the Stockroom, and in the latter a proposed interaction with SPRUCE.

1. Query S3
2. S3 returns artifact
3. Deploy S3 artifact

Figure **cenario**

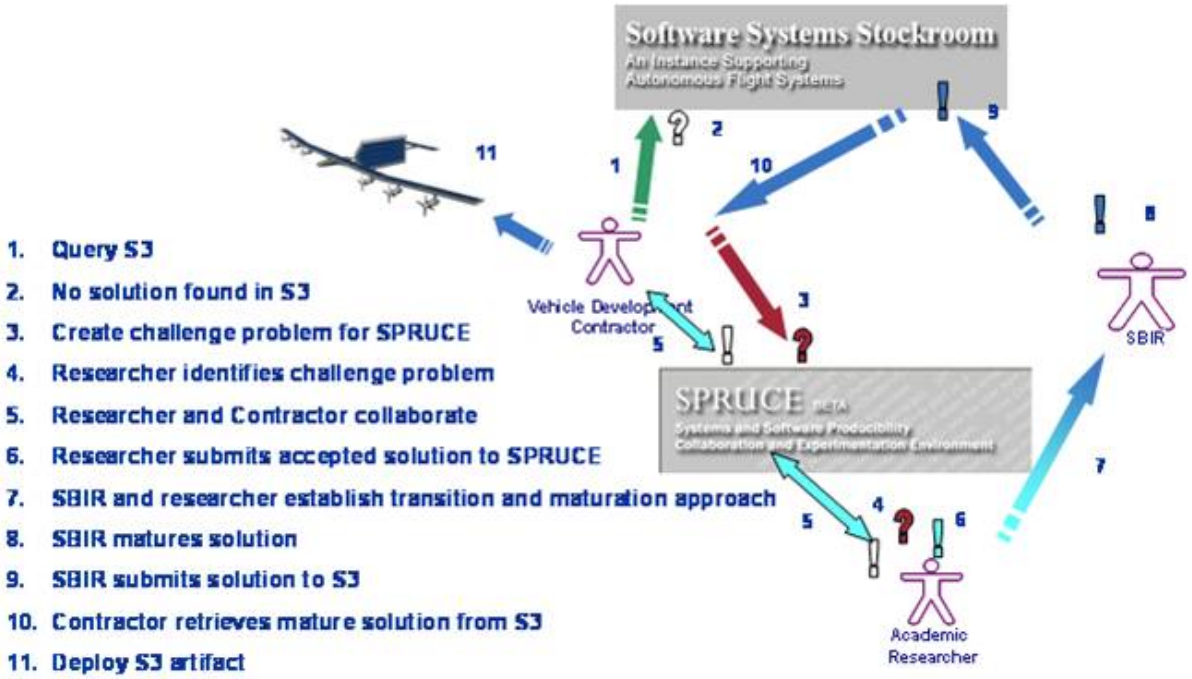


Figure 17: Unsuccessful Stockroom Search Followed by SPRUCE Collaboration

Figure 16 is essentially the graphical equivalent to the sequence diagram shown in Figure 15, where a Vehicle Development Contractor searches the Stockroom for a technical solution. The Stockroom is able to provide an artifact with acceptable properties, which is ultimately deployed.

Figure 17 demonstrates the case where no solution is found on the Stockroom. Events 3 through 8 depict a subsequent interaction with SPRUCE, in which a challenge problem is created to address the contractor need. The sequence of events goes on to show a successful engagement with the research community, followed by technology transition and maturation (via SBIR engagement). Events 9 through 11 demonstrate how the technology developed and incubated in SPRUCE could then be submitted back into the Stockroom (or possibly just the metadata description), in order to support the domain specific technology search with semantic web support available on the Stockroom.

4.2. As-Built Configuration of Prototype Stockroom and Contents

The following sections provide a detailed description of the S3 prototype as built during Phase 1, including a summary of S3 capabilities, the resulting S3 architecture, and a summary of S3 domain contents.

4.2.1. Stockroom Capabilities

The purpose of the stockroom is to preserve and propagate knowledge in the domain of the development of embedded autonomous systems flight software on emerging processing systems. All of the stockroom capabilities have been designed with this end in view. The stockroom Phase 1 prototype implementation has been developed both to provide initial capabilities and to prototype capabilities in support of implementation and operation of the stockroom in Phase 2. Likewise, the prototype stockroom contents have been selected both for the domain knowledge they encapsulate and their value in supporting Phase 1 experimentation and prototyping activities.

The use cases described in Section 4.1.2 were used to identify the stockroom capabilities required to support the stockroom purpose. As with the use cases, the capabilities provided by the stockroom will be treated in three categories: artifact centered capabilities, collaboration capabilities, and administrative capabilities. The remainder of this section provides an overview of the stockroom prototype capabilities in these three areas. Specific details of the capabilities and their implementation are covered in Section 4.2.2, while details about their use and configuration are available in the “User’s Guide for S3 Operation” [6] and the “Administration & Installation Guide S3 Operation” [7].

4.2.1.1. Artifact Centered Capabilities

Artifacts are the vehicle for capture and re-use of domain knowledge. The domain knowledge associated with an artifact includes the artifact name, a list of keywords, a description, optional parent artifacts, an optional attachment, an optional URL, and metadata. The metadata includes attributes such as artifact type, keywords, license, author, security and export classifications, as well as triples that specify RDF-like relationships between artifacts and taxonomy terms. Artifacts, particularly those that correspond to software entities such as tools or libraries, may

have an associated payload (e.g., the source files for a library) in the form of (i) an attachment if the payload is actually stored in the stockroom, (ii) a URL if the payload resides at some other location, or (iii) a point of contact or other information that identifies how the payload may be obtained. Other artifacts, particularly those that correspond to more abstract types of domain knowledge (e.g., the concept of a UAV), may have no associated payload. An artifact is represented to the stockroom user as a web page.

The stockroom provides capabilities for adding artifacts and information about artifacts to the stockroom. This corresponds to preserving domain knowledge. Stockroom users may add artifacts to the stockroom by creating an artifact web page, supplying the appropriate information defining and describing the artifact, including its payload if any. Stockroom users may also add metadata about existing artifacts by editing the web page associated with the artifact, or by defining predicates and triples that apply to the artifact.

The stockroom also provides capabilities for identifying and retrieving artifacts and information about artifacts from the stockroom. This corresponds to the propagation of domain knowledge. The stockroom provides four capabilities for identifying artifacts: browsing, natural language search, metadata search, and ontological search. Once an artifact has been identified, it may be retrieved by viewing its web page, and then using the appropriate mechanism for retrieving its payload, if any. For artifacts hosted by the stockroom this would simply involve downloading the payload attachment directly from the stockroom artifact web page. For artifacts with a URL supplied for the payload, information about acquiring the artifact can be found by following the URL. Artifacts with payloads identified in other ways (e.g., by a designated point of contact) can be acquired by following those artifact specific instructions and processes.

In addition to the capability to browse the entire collection of artifacts, either as a complete list or by means of one of several “tag clouds”, the stockroom provides three types of searches. Natural language search corresponds to a “Google™ like” whole text search of the entire stockroom contents (including non-artifact content). The result of a natural language search is a web page with links to stockroom web pages on which the search term(s) appear. The other two search types only search information associated with artifacts. Metadata search, which includes a guided search capability, identifies those artifacts that match the search, and such searches can be constructed incrementally. Metadata search also includes a guided search in which the search terms are restricted to terms from the ontology and taxonomies. Ontological search is provided by the stockroom as the capability to execute description logic queries against an ontology capturing all of the metadata of the stockroom. This includes the capability to reason over artifact attributes and relationships among artifacts.

4.2.1.2. Collaboration Capabilities

Collaboration capabilities are provided to support the direct propagation of domain knowledge between stockroom users to supplement the indirect propagation provided by stockroom artifacts, and to support and nurture the stockroom community. The collaboration capabilities are primarily exercised through participation in WGs. In Phase 1, we identified two main types of WG that would be centered around the stockroom and the stockroom community: TWGs and CSWGs. TWGs would be focused on aspects of a particular technology, which could be an abstract technology area, or a particular artifact or set of artifacts in the stockroom.

CSWGs would be focused on developing a community consensus in a particular area, in particular, developing a stockroom community "standard" for a particular technology. Note that the CSWG would not be considered an official standards body, nor would its product an official standard, though it may be desirable in some instances to submit the product of a CSWG to such an organization.

The primary collaboration capability used during Phase 1 by WGs is a file sharing capability provided by the Drupal Web File Manager module. As configured in the Phase 1 prototype, this provides each working group with an access controlled file sharing mechanism. It provides a hierarchical work space to each working group, in which subfolders can be created to organize WG content, and in which files can be uploaded for sharing with the working group. Other collaboration mechanisms are also available, including blogs, fora, news items, and web pages, though these were not used extensively in the Phase 1 WGs. All of these mechanisms can also be made available for non-WG use by stockroom users, and some may be used for dissemination of stockroom news and administrative information to the stockroom community.

4.2.1.3. Administrative Capabilities

The administrative capabilities of the stockroom are provided to ensure availability of and access to the domain knowledge contained in the stockroom. To that end, there are four major administrative capabilities provided by the stockroom. Account management capabilities provide access to the stockroom domain knowledge by authorized users. Stockroom maintenance capabilities ensure the availability of the stockroom content. Ontology and taxonomy management capabilities ensure the continuing relevance of the stockroom and that the taxonomies and ontology continue to support precise access to stockroom domain knowledge. Finally, the stockroom metrics capabilities provide data to guide the evolution of the stockroom.

There are three main account management use cases that are served by stockroom administrative capabilities. The stockroom administrators can create user accounts. This includes both the act of creating the actual stockroom Drupal account for a new user, as well as making the determination to grant access to the stockroom, in accordance with stockroom access policies and, if required by those policies, in consultation with the stockroom sponsors. The stockroom administrator will also maintain user accounts. Account creation and maintenance includes setting and adjusting each account's role based access settings. In addition to creating and maintaining individual accounts, the stockroom also provides capabilities for creating and maintaining the various stockroom working groups.

The second set of administrative capabilities centers on maintaining the stockroom, in particular, the artifact content that is the core of the stockroom domain knowledge. This includes maintaining the artifact content and performing backups. The Phase 1 stockroom has been designed to include automated backups of stockroom content. Other stockroom maintenance activities are included in the artifact management workflows, which cover activities such as approving stockroom contributions and support for rolling back to previous versions of artifacts if necessary.

As further discussed in Sections 4.2.2 and 4.2.3, especially Section 4.2.3.3, the taxonomies and ontology that enable efficient retrieval of stockroom domain knowledge artifacts are not static structures determined at stockroom design time. Rather, they are dynamic and evolving snapshots of a rapidly changing domain. This stockroom concept, including the Phase 1 stockroom prototype, therefore provides a number of mechanisms to support the evolution of the stockroom taxonomies and ontology. This includes both changing the taxonomies and ontology themselves, as well as the migration of the existing stockroom artifact metadata.

Finally, the stockroom administrative capabilities include the ability to collect and analyze a variety of metrics about the stockroom content, the stockroom community, and the ways in which the community makes use of the stockroom. This information provides data to support all of the other stockroom activities. For example, metrics on the size of the keywords taxonomy and the number of artifacts returned by each search could be used to drive the evolution of the stockroom ontology and taxonomies, while information about user profiles and network traffic could be used to direct community building efforts.

4.2.2. Architecture

The Software Systems Stockroom for autonomous flight systems builds upon relevant standards and open-source technologies to create a suite of web-based applications, which provides the infrastructure for a virtual community for the collection and dissemination of information relevant to complex fields such as software engineering for autonomous air vehicles.

Accessing the Stockroom by way of a web browser, geographically dispersed individuals come together to form communities of interest, and to capture and reuse domain knowledge and expertise through an open, community-driven, technically focused shared infrastructure.

At its heart, the Stockroom application is built upon a well-structured stack of application service layers built upon a web server and a content management system (Figure 18). This overall architecture is described below.

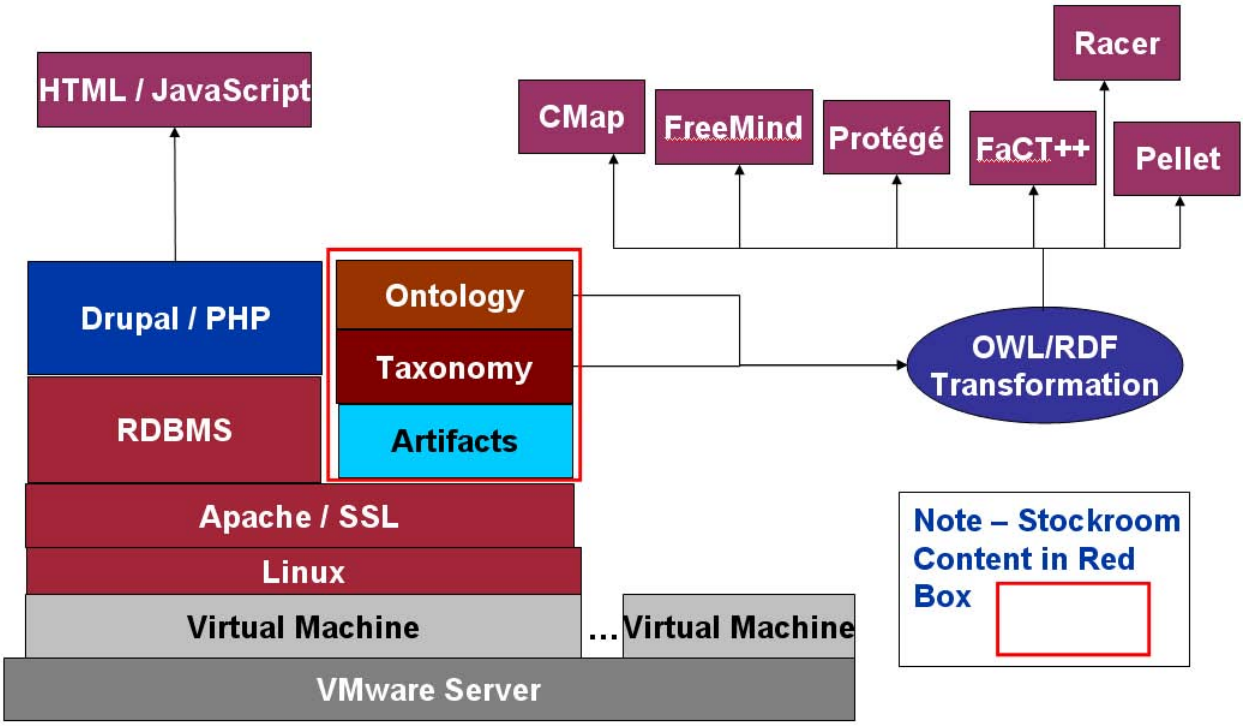


Figure 18: Stockroom Notional Architecture

In the following section on hardware and systems software, we begin with a quick overview of the lowest, most generic layers of the system. Continuing from there, we work our way up to the unique features of the Stockroom system.

4.2.2.1. Hardware and Systems Software Infrastructure

The Stockroom itself relies on a great deal of pre-built components and frameworks necessary for composing the final system. In this section, we review the critical off-the-shelf applications that are leveraged to form the underlying foundation upon which Stockroom is built.

4.2.2.1.1. The Stockroom Network Appliance

Stockroom is delivered as a network appliance on a removable storage device such as Digital Video Disc **(**DVD) media. Essentially, this means that the stockroom product takes the form of a virtual machine (VM) that contains an operating system, a web server, the content management system and all other software layers that constitute the Stockroom. The VM exists in the form of a set of large files that can be 'run' under a host operating system. Using virtualization technology, the Stockroom application and its requisite systems software have been cleanly separated from the hardware and host operating system that runs S3. As delivered, the prototype VM is compatible with the *VMware* standard for virtualization, and thus the S3 virtual machine may be played within any VMware compatible tool such as "VMware Player" [8] or "VMware Server"[1].

4.2.2.1.2. The Stockroom's Systems Software

Booting the S3 VM under VMware Player will start up the virtual machine's operating system, which will then proceed to initialize several key pieces of systems and applications software that Stockroom is dependent on.

The operating system deployed for the initial prototype of S3 is Ubuntu Linux. However, the choice was somewhat arbitrary. Ubuntu is a popular, free, open-source variant of Linux, but any OS supported by the virtualization technology, which is also compatible with the required software described in subsequent sections, will also support an instance of the Stockroom. In addition to the dozens of alternative flavors of Linux, this also includes the Microsoft Windows family of operating systems. Please see the Administrator's Guide [7] for a precise enumeration of the various operating systems supported at the time of this writing.

Being a web-based application, S3 obviously requires a web server to fulfill incoming HTTP requests. As delivered, the initial prototype of S3 relies on the de facto standard Apache web server. Alternatively, in a Microsoft Windows environment, one may choose between Apache and the Microsoft IIS product.

Not only is S3 a web-app, but it is a *database backed* web application. In accordance with the model view controller (MVC) design pattern, the data of the Stockroom is cleanly separated from any presentation layers that provide the user his interface or *view* into the data. All Stockroom data is therefore stored in a relational database management system (RDBMS). The RDBMS chosen for the S3 prototype is MySQL [9] due to its particular suitability for fronting database-backed websites. However, any RDBMS compliant with the Drupal content management system is suitable.

S3 is developed primarily in the PHP programming language, and therefore requires PHP5. Additionally, some ancillary Stockroom code is written in the Java programming language. Both PHP and Java have been pre-installed on the VM's instance of Ubuntu.

---

[1] Alternatively, tools exist to convert VMware virtual machines to other virtualization tool providers' file formats.

4.2.2.1.3. The Content Management System

Stockroom is built upon the Drupal content management system (CMS). In general, a CMS is an application that provides the capability for collaborative creation, publication, navigation, and search of electronic media, and is accessible via a web server.

In particular, Drupal is a CMS system focused on configurability and extensibility. Out of the box, Drupal can be easily configured to perform most commonly accepted, web-based interaction paradigms such as wikis, news item publication and announcements, discussion forums, personal web blogs, and static, corporate/organizational web sites. In addition, however, Drupal provides a full-fledged development framework. Via well thought out abstractions and plug-in points, Drupal may be customized and extended to provide unique capabilities and novel services.

In the remainder of this section the common, Drupal's out-of-the-box features are summarized. Following that, Section 4.2.2.2 presents the unique, custom engineered manifestations of Drupal that comprise the Software Systems Stockroom.

The most fundamental concept in Drupal is that of a *Node*. A Node in Drupal subsumes the idea of a *web page* in the traditional world of Hyper Text Markup Language (HTML). At a minimum, every Node has a *name* and a *description* attribute. These roughly translate to the <title> and <body> sections of a traditional web page. Furthermore, Nodes have been specialized in Drupal to provide a type system of specific content types available (Figure 19). For example, *News Items* may have an additional attribute that specifies an *expiration time*, at which point the News Item is scheduled for archiving.

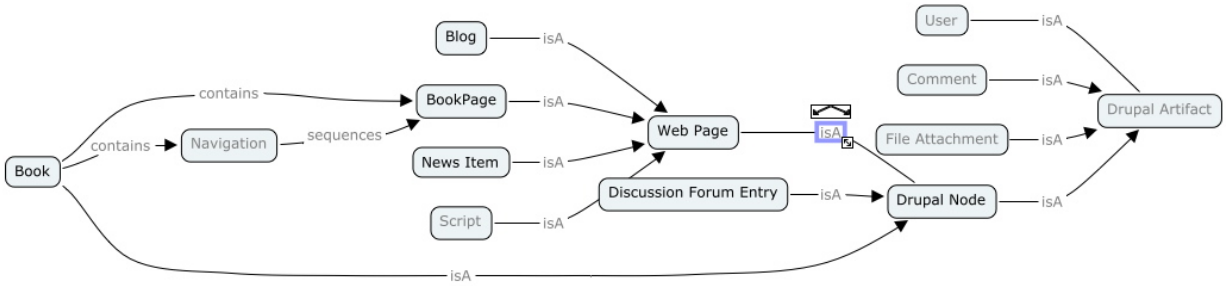


Figure 19: Drupal Node Hierarchy

Stockroom uses many of these predefined, specialized content types as part of its *electronic collaboration* features. For example, TWGs typically have one or more discussion fora set aside for members to carry on threaded, virtual conversations. Additionally, web pages may be aggregated together and supplemented with pre-defined navigational constructs to form the equivalent of an electronic Book. Another commonly recognized web-based content type, which Drupal supports, is the *web log* or Blog. Figure 20 shows an example of one user's blog on S3.

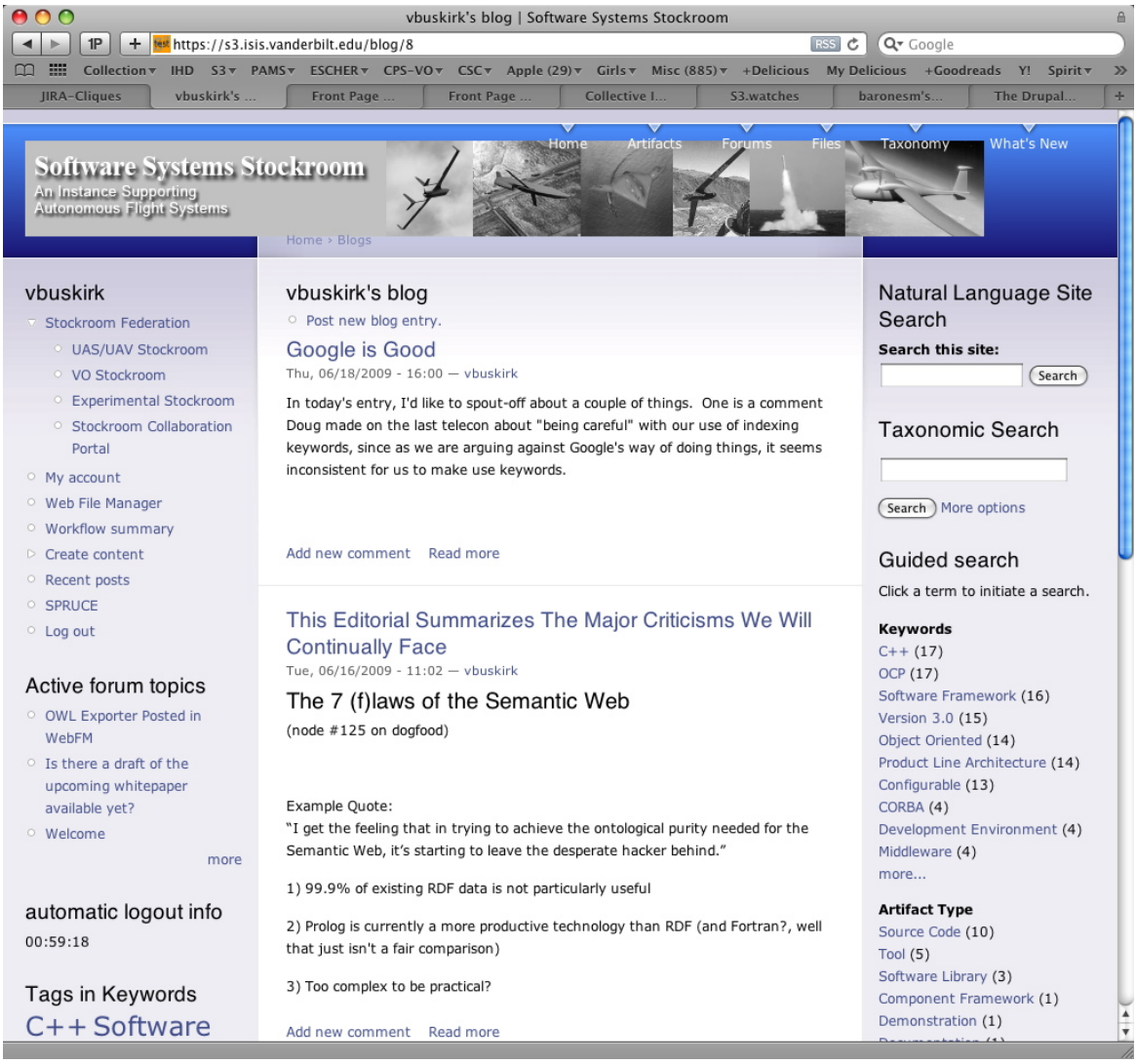


Figure 20: Sample S3 Blog

In addition to the pre-defined libraries of Node types in Drupal, there also exist first class concepts for representing Users and also for Attachments and Comments, which may both be associated with existing Nodes.

As mentioned previously, Drupal is especially geared toward the goals of configurability and customization. It therefore has very powerful facilities for extending the behavior of the website.

First is the ability to easily define arbitrary, new content types. Along with the name of the new content type, one specifies what attributes are to be associated with content of this type, in addition to the default *name* and *description* fields, which all Drupal Nodes have. For example, a new content type called Car might be defined, which has among other attributes a Manufacturer, Model and Color attributes. Upon creating any new Nodes of the type Car, a web form would automatically be provided for specifying the value of these three fields. Attributes may be specified as required or optional, with single or multiple cardinalities.

The content-specific fields described above have data types. They may either be primitive data types such as Strings or Integers, or one may define new data types that are bound to a Drupal *Taxonomy*. A taxonomy is an organized set of concepts that classifies content – taxonomies are first-class content elements in Drupal. In the Car example from above, we might choose to allow the user to type in free form text for Color (e.g. "plum"), or we could enumerate all of the acceptable values for Color using the *Taxonomy* constructs of Drupal. In the later case, the valid options might be {red, orange, yellow, green, blue, & purple}[2]. Additionally, taxonomies may declare hierarchical relationships. A Ford Motor Co. vehicle taxonomy might categorize the F-150 and the Ranger as Trucks, while the Mustang is a Sports Car. Taxonomies are covered in detail in Section 4.2.3. At this point, the key idea is that by using Drupal's facilities to create custom content, the ability is provided to apply a type system to the types of information that are typically represented in untyped web pages. This type system is the key to programmatically manipulating knowledge.

Drupal provides a well-defined set of plug-in points to inject custom code dynamically. These Drupal *Hooks* are effectively events at which one might wish to strategically modify the behavior of the running system. For example, Drupal provides a hook named *hook_delete* that is related to the user-initiated act of deleting a Node. By following a simple naming convention for custom code, the events are specified that the code will then execute. Registering a method[3] with Drupal whose name ends with the string "*_delete*", for example, would result in Drupal calling this method (with a pre-determined set of input parameters) each time any user attempted to delete any existing Drupal Node. Our code could then override or supplement the default behavior for deleting Nodes (e.g. to make a backup copy, to email an administrator, or to apply heightened security for certain types of Nodes).

The look and feel of a Drupal site can be easily customized with its *theming* facilities. It is even possible to make the look and feel of a Drupal site vary between individual users. The screenshots of

Figure 21 and Figure 22 show two of the themes used during the prototyping Phase of S3.

[2] Note that taxonomies may be run-time extensible, as opposed to being required to be fully enumerated ahead of time.

[3] In Drupal, custom code is implemented using the PHP programming language.

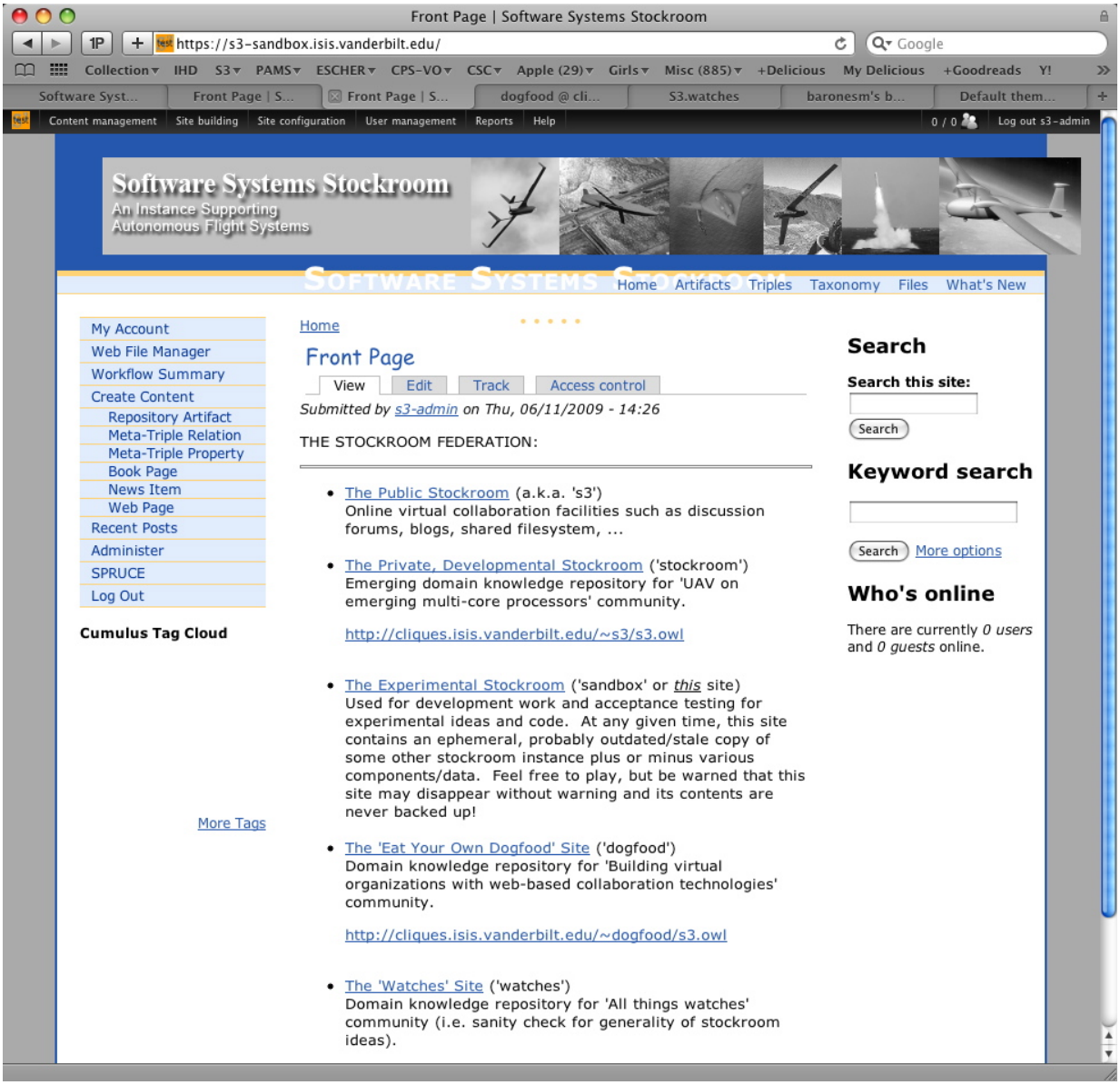


Figure 21: S3 Theme

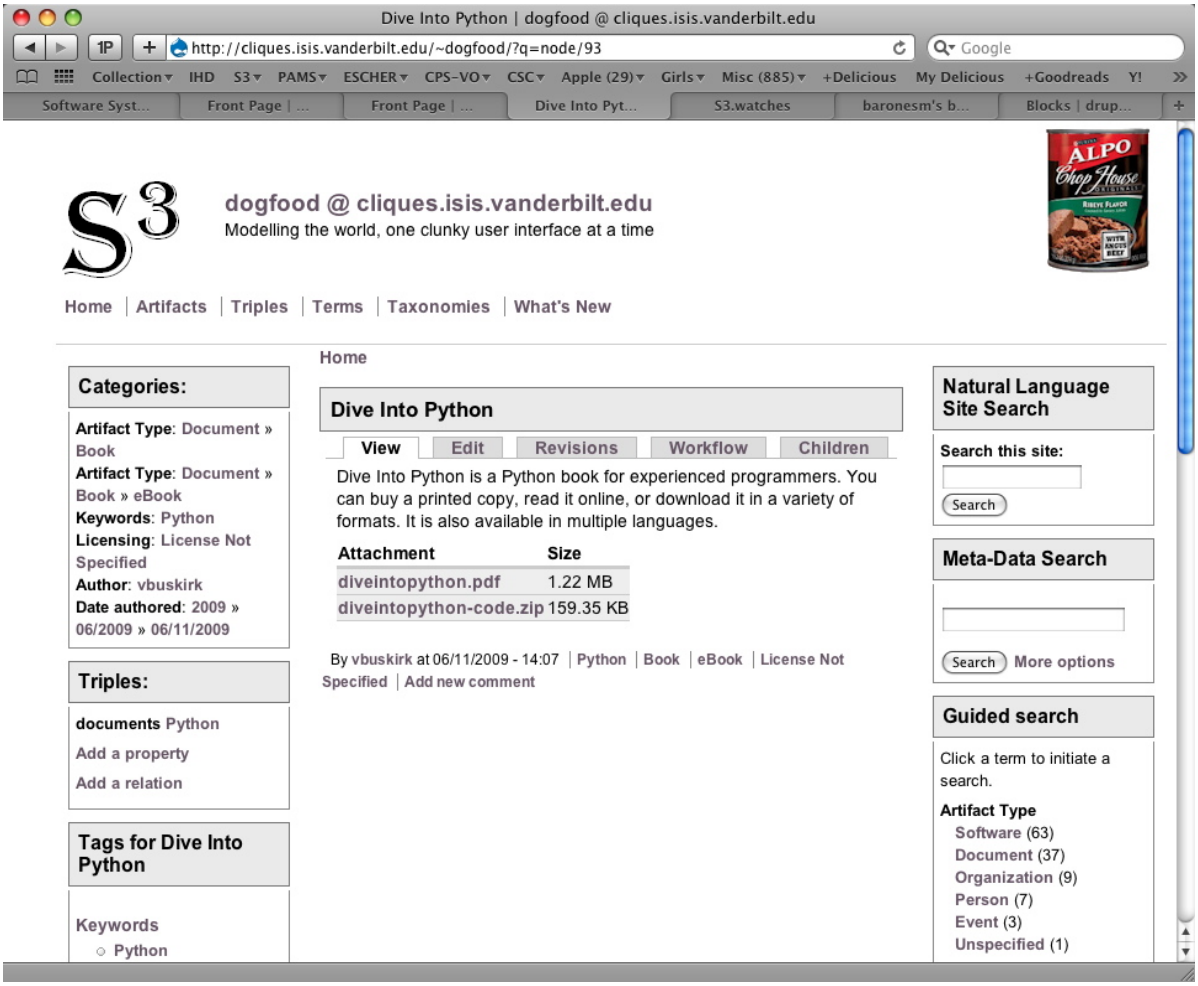


Figure 22: Alternative Drupal Theme

In addition to the primary content area, Drupal themes typically divide the page up into sidebars, footers and headers. *Blocks* are Drupal's mechanism for injecting custom code into these sidebars. In the Figure 22, there is a custom block named 'Triples' along the left sidebar, which tells us that the electronic book named Dive Into Python documents the Python programming language. This Triples block is an example of custom code developed during the S3 prototyping phase.

Finally, *Modules* are Drupal's terminology for a collection of related files containing some PHP code. Like Blocks and Themes, Modules relying on the Hooks architecture to inject their behavior into a running Drupal system. Unlike Blocks and Themes, Modules aren't constrained in their focus (i.e. sidebar navigations or presentation layer layout). Drupal is a set of Modules. The *core modules* are provided with the initial download. Any additionally customized programmed behavior will be packaged up into site-specific Modules and installed on the running Drupal instance.

4.2.2.2. The Stockroom Application

The Stockroom application is a collection of Drupal modules that are configured according to the goals of the project, plus a set of additional tools that together implement various functionalities. Below are described the application in the form of these functionalities, indicating the (1) Drupal modules used and configured, and (2) the custom code built to provide the functionalities.

As shown on Figure 18 the Stockroom is a highly customized content management system that carries the primary content in the form of Stockroom artifacts, the taxonomy and the ontology. External tools that provide services for ontology management and semantic search are connected to the Stockroom via an OWL/RDF bridge that allows bringing the power of state-of-the-art semantic web tools into the system.

4.2.2.2.1.User Management

The goal of user management is to control what a specific kind of user is allowed to do with the Stockroom. The prototype implementation accomplishes this through the use of access control lists (ACL's) that support role-based access control. Users must register with the Stockroom to gain access and when registered they are assigned one or more roles. Access and modification to every resource in the Stockroom is permitted or denied based on the role of the user.

4.2.2.2.1.1. Roles

The overall design of the Stockroom assumes a set of roles listed below.

- Consumer: A user who reads, who searches for, and downloads content.
- Contributor: A user who creates and contributes content to the Stockroom.
- Manager: A user who can execute management decisions on the site (e.g. changes in the taxonomy).
- Moderator: A user who can moderate (approve or disapprove) new content posted to the Stockroom.
- Administrator: The person who has ultimate authority over the Stockroom and can configure every detail on it (including its design and implementation).

In the prototype Stockroom we have implemented the above scheme partially to better reflect the momentary needs of the development. The following roles are implemented:

- Anonymous user: A user who has no access to the Stockroom, except the front page that informs him/her about what the Stockroom is for and how to request access.
- Authenticated user: A user who has logged into to the Stockroom and has rights to create and edit content.

- Admin-taxonomy: A user who can administer changes to the taxonomy contained in the Stockroom.
- S3-team: A user who can make design changes on the Stockroom (member of the design team).
- Admin-webfm: A user who can administer changes to the virtual file system of the Stockroom (see below).
- Admin: Same as the Administrator from the design above.
- Individual TWG Role: These users have rights to access specific parts of the virtual file system containing files for specific working groups, detailed below.

The above implementation is considered temporary and will be refined in the next phase.

4.2.2.2.1.2. Working Groups

Early in the design of the Stockroom the requirement arose for supporting collaborative work through a simple file exchange mechanism. Using the virtual file system module discussed below, a file repository was created where files could be stored and manipulated using a web browser. The repository is segmented into separate folder hierarchies that the members of the corresponding user groups, called 'Working Groups', can access. Each Working Group has a corresponding role, and when a user registers s/he will be given a role in one or more Working Groups. Each Working Group addresses a particular need in development of cyber-physical systems.

The following Technical Working Groups are currently configured:

- Education TWG
- Formal Methods TWG
- High-Confidence Design TWG
- MCAR TWG
- Transportation TWG.

Additional information about these TWGs may be found in Section 4.2.4.2. Additional Working Groups were proposed and will be set up in the final Stockroom product.

CSWGs: These working groups are formed to develop and evolve standards in the community. For example, a working group devoted to developing standards for UAV control API-s would belong here.

The Ontology Evolution Steering Committee (OESC): This CSWG deals with designing and evolving the taxonomy and ontology of the Stockroom. The taxonomy and the ontology used in the Stockroom is not static, rather it should evolve as new content is added. The OESC is responsible for managing and facilitating this evolution. See Section 4.2.3.3 on taxonomy and ontology evolution for further details.

4.2.2.2.2.Custom Content for Stockroom

The purpose of the Stockroom is to provide content – software engineering artifacts for the application domain of flight software for unmanned systems. As Drupal built-in content types are very simple, we have defined dedicated, domain-specific new content types for the Stockroom. As some content is inherently file-based, we have used and configured a virtual file system.

4.2.2.2.2.1. The Virtual Filesystem

In addition to the typical, web-based content types available in Drupal, the Stockroom has been supplemented with the WebFM web file manager add-on module,

Figure **23**. WebFM provides an Asynchronous JavaScript And XML- (AJAX-) enabled virtual file system for storage of arbitrary artifacts organized within folder hierarchies.

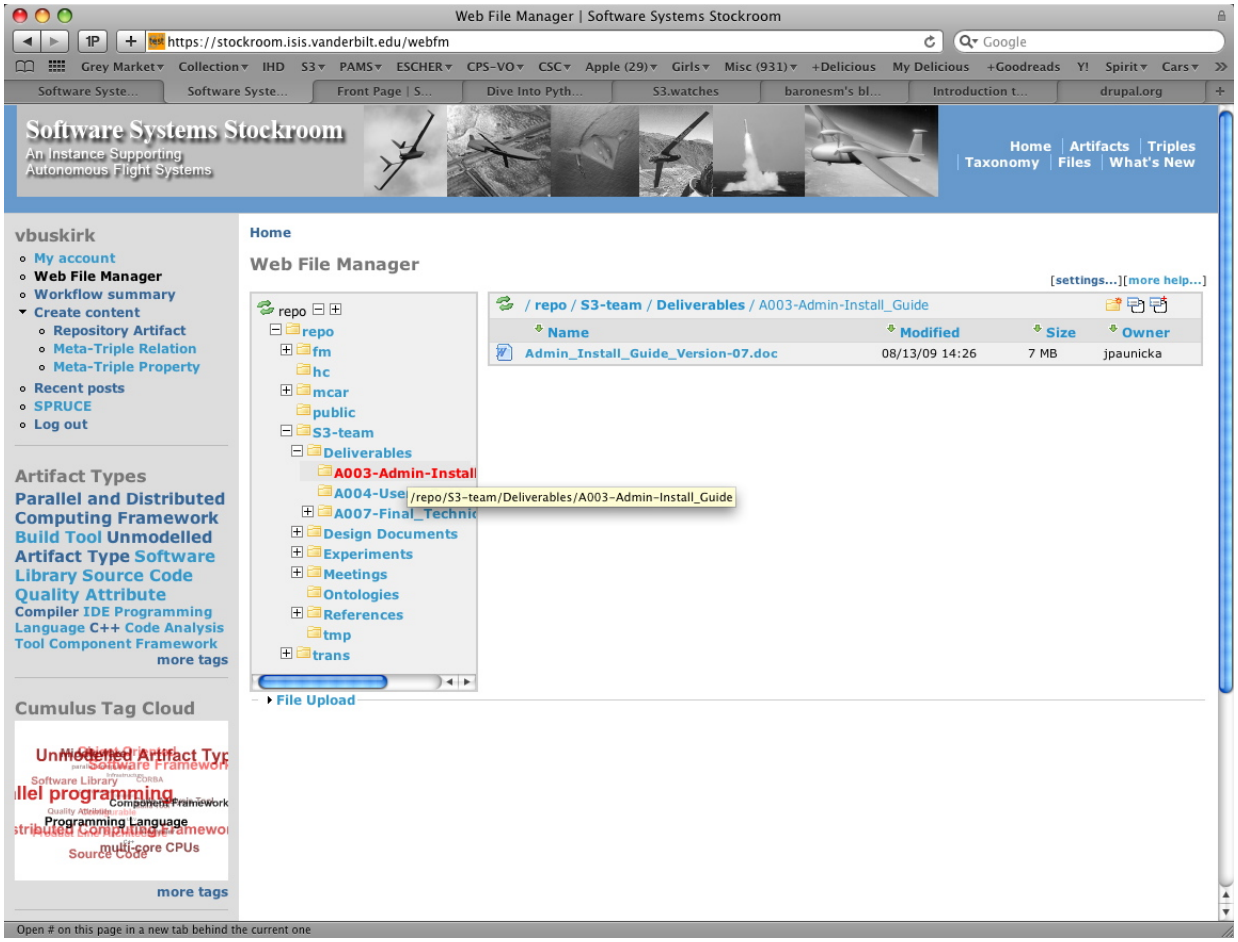


Figure 23: Web File Manager

WebFM is implemented as a Drupal module, and therefore may be enabled/disabled and configured by way of the usual administrative facilities of Drupal. Specific details of how WebFM is configured to support role-based access are provided in the later section on technical working groups (TWGs). For example,

Figure 24 shows that WebFM may be turned on or off using the standard *Site building* section of Drupal's administrative pages.

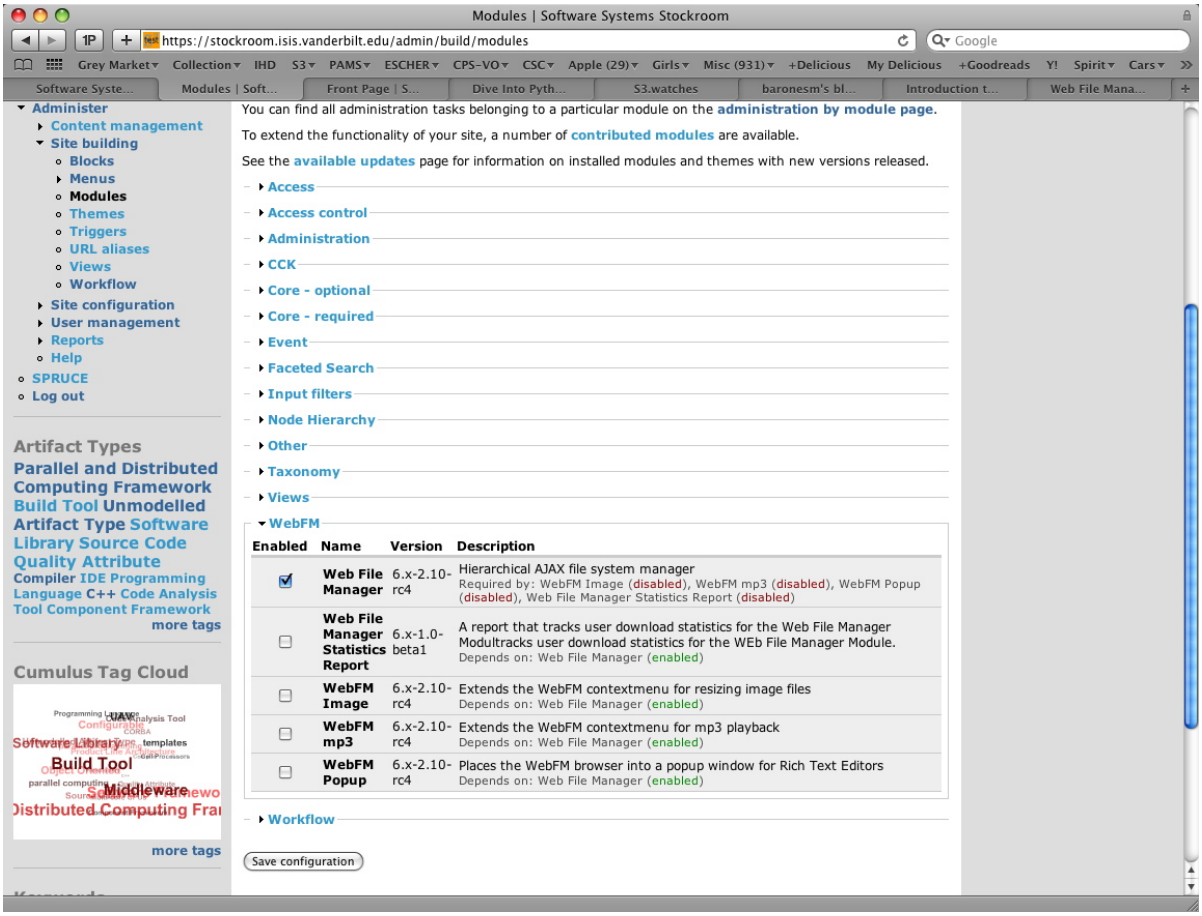


Figure 24: Enabling/Disabling Web File Manager on For an Individual WG

4.2.2.2.2.2. Stockroom Artifacts

As mentioned previously, all Nodes in the Drupal CMS have a *name* attribute and a *description* field. These fields are equivalent to the commonly found <title> and <body> sections of a standard HTML web page. Additionally, for the purposes of the Stockroom application, we have designed and implemented customized data types for representing Stockroom content. The principle customized content type is termed the *Repository Artifact*. These Artifacts are distinguished from typical web pages by the addition of several unique facets intended to store metadata about entities relevant to the UAV domain.

In addition to *name* and *description*, each Stockroom artifact page is annotated with a record of the authorized user who created it, as well as the creation date. Furthermore, as Artifacts are modified over time, the author and timestamp information for each editing operation are recorded (as well as the specific differences between revisions).

For prototyping purposes, we modified the *User* data structures of Drupal to also track the *Organization* that each user is associated with. Therefore, individual Artifacts may be associated with not only people, but with companies also. This additional organizational information is presented as a proof of concept; similar extensions could be implemented as needed in Phase 2.

Related to the notion of pedigrees are the related concepts of workflow management and revision tracking. As indicated above in the section on Roles, there is a class of users in the Stockroom who have been designated as moderators. Moderators are responsible for helping to maintain the high degree of fidelity and relevance of the information contained in the Stockroom knowledge base. As new information comes into the stockroom, it is flagged as a candidate for moderation. As moderators review the content of new Stockroom information, they may reject the material in question as unsuitable, recommend it for approval with minor modifications, or simply approve it. In the event of rejection, the moderator may use the revision tracking system of Stockroom to *roll back* to a previous version of the artifact (i.e. effectively deleting the latest edits). This moderation process itself contributes to the pedigree of individual Artifacts. Those Artifacts that have successfully passed the moderation process are obviously more trusted than those that have not[4].

Every Stockroom artifact is explicitly associated with its designated security classification (e.g. *Unclassified*). Similarly, there are markings available for ITAR restrictions, as well as for documenting EAR classification. These attributes are required to be specified before any new artifact will be considered for inclusion into the Stockroom.

Additionally, there is an optional area for specifying the licensing terms for the subject of some Stockroom metadata. This licensing attributed is tied to a complex Drupal taxonomy, which documents all of the known licensing agreements recognized by Stockroom (see the taxonomy sections below for more details). While quite complete, this taxonomy is of course extensible.

For Stockroom, Drupal Artifacts have been extended so that they may have parents and/or children. While optional, this notion is often useful for describing certain containment relationships. In the screenshot of Figure 25 below, we see that OCP comprises several smaller, individual artifacts.

[4] Note that it is possible for Artifacts that have not yet met moderator approval to be hidden in an *unpublished* state, thereby being visible to only the original author and to the moderators group. This feature is presently disabled.

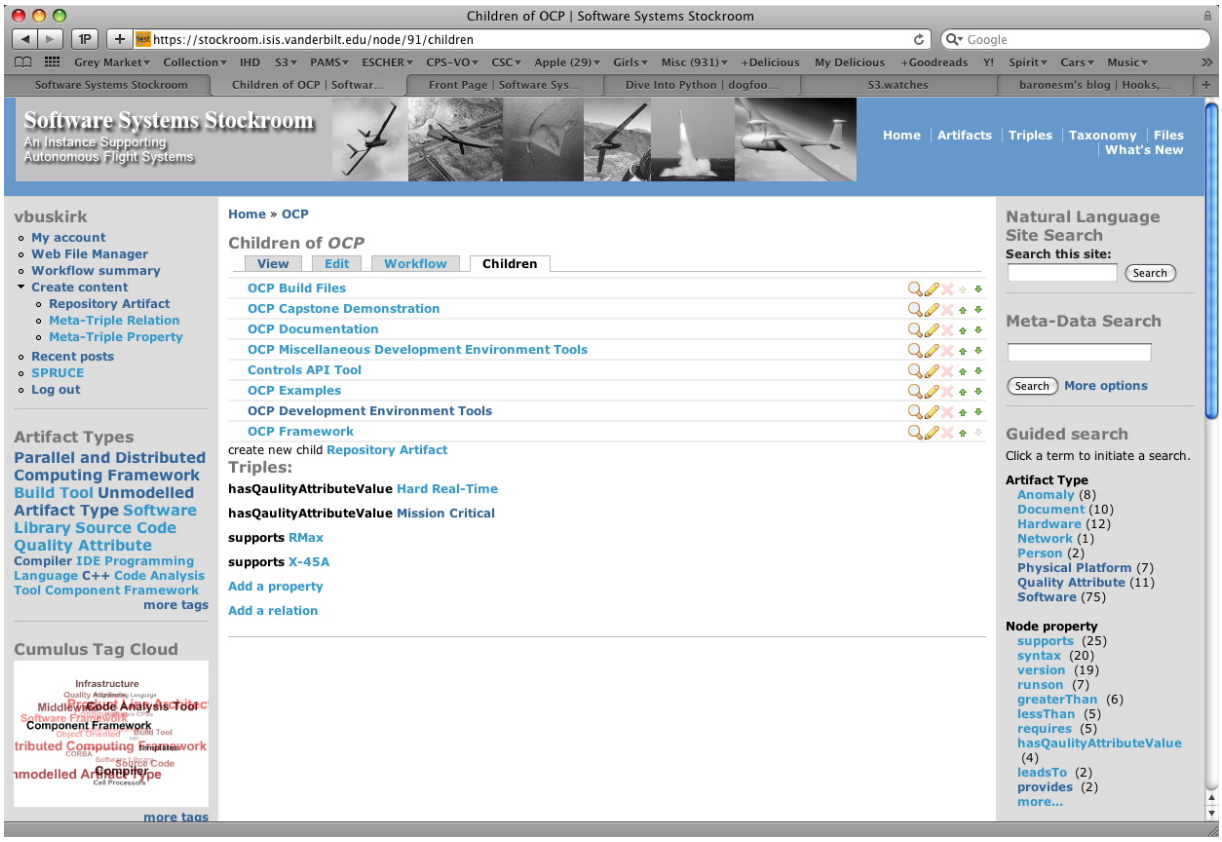


Figure 25: Displaying the Children of an Artifact

Each Stockroom artifact has an optional *URL* field that may be used to reference an external source for the target payload. As Stockroom is intended primarily to house metadata, we do not necessarily require that the object which is the subject of this metadata, be housed within the Stockroom itself[5]. The URL attribute, for example, could point to some third-party vendor's designated web page for purchasing license or evaluation copies of some software component. Note that if the URL attribute is used, the string provided is validated to ensure that is conforms to the standard uniform resource locator syntax.

In the event that it is appropriate to store the actual payload in Stockroom, we also provide for this by way of zero or more File Attachments. When adding or editing an artifact, a section of the associated HTML form is dedicated for managing these attachments (including the ability to browse your local file system in a pop-up dialog when specifying a new Attachment).

[5] That is, we distinguish between artifacts, the metadata and the payload.

Every artifact in Stockroom is associated with a *type*, and this type field is bound to a taxonomy that describes the family of artifact types supported in Stockroom. Later sections discuss the details of taxonomies and how they are used. For now, just consider that Artifacts are further specialized into categories such as Documents, Events and Software, and that Documents, for example, are further specialized into sub-categories such as Books, Scholarly Journal Articles, and Users' Manuals (which are then further refined). This provides a rich type system for artifacts.

Interacting with the type system described in the previous section is the Triples sub-system. Whereas until now we have been discussing *statically* defined attributes, the Triples facility effectively allows the incorporation of dynamic typing into Stockroom.

A Triple is defined as a 3-tuple '(subject, verb, object)', where the first and last slots are filled by references to Stockroom elements. The second slot is tied to another Stockroom taxonomy called the Predicates taxonomy. Using the triples notation we could say, for example, say that some electronic book 'documents' some programming language. In fact, in the prototype instance of Stockroom, the OCP Documentation is declared to describe the OCP framework using an `('OCP Documentation', documents, OCP)` triples. See the triples Block[6] in the footer of the content pane in the following screenshot.

[6] Remember that *Blocks* are Drupal's term for custom engineered code packages, which display themselves graphically within the border areas of web pages. In this case, the Block named "Triples:" is displayed in the footer area.

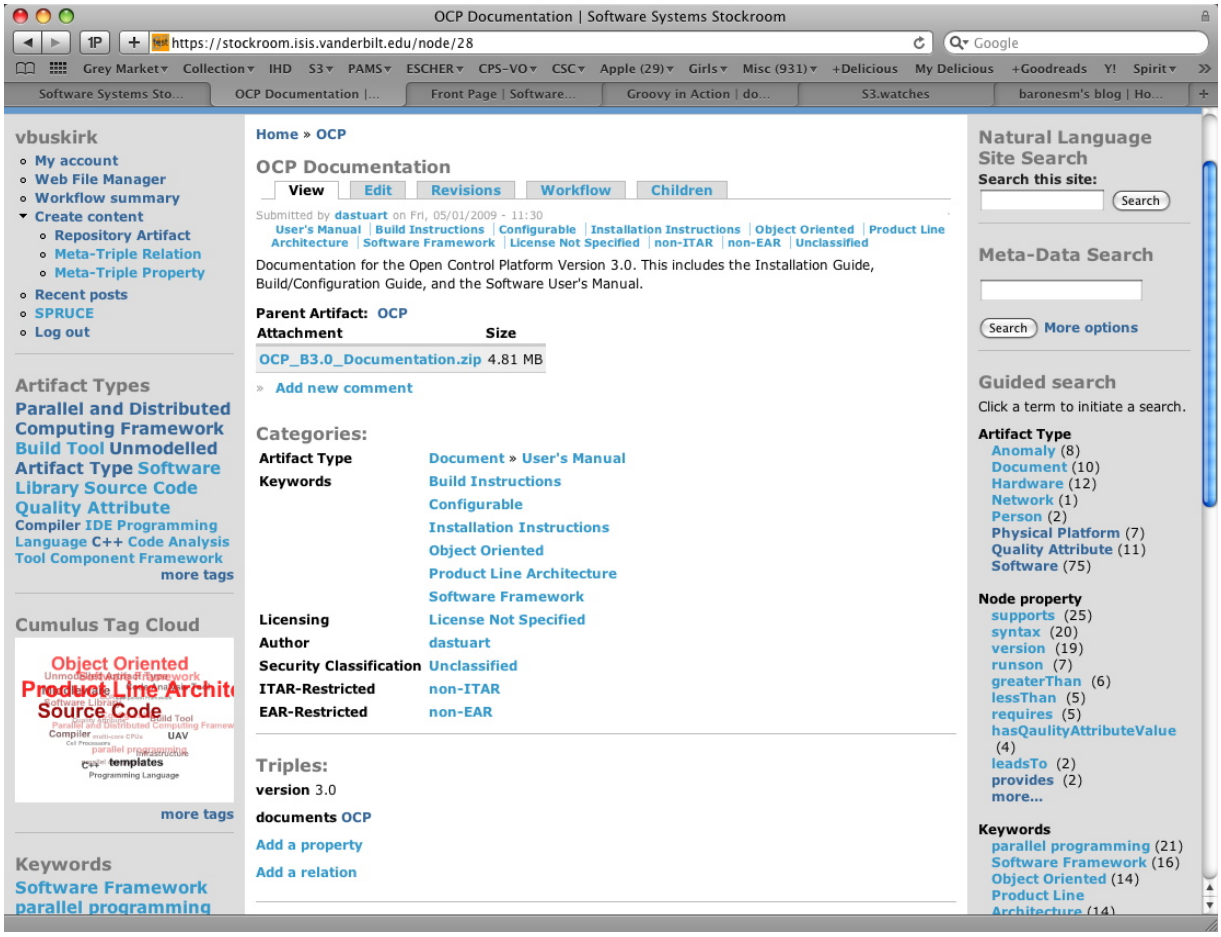


Figure 26: Triples Associated with an Artifact

While in this example, that fact is implicit in the title of the 'Open Control Platform (OCP) Documentation' artifact. By formalizing this knowledge into a Triple, we enable Stockroom itself to become self-aware. We may then, for example, use symbolic reasoning techniques to discover all documentation relevant to a particular technology. In the following screenshot (Figure 27) we see that both SPP and Gedae are development environments intended to support the IBM Cell processor.

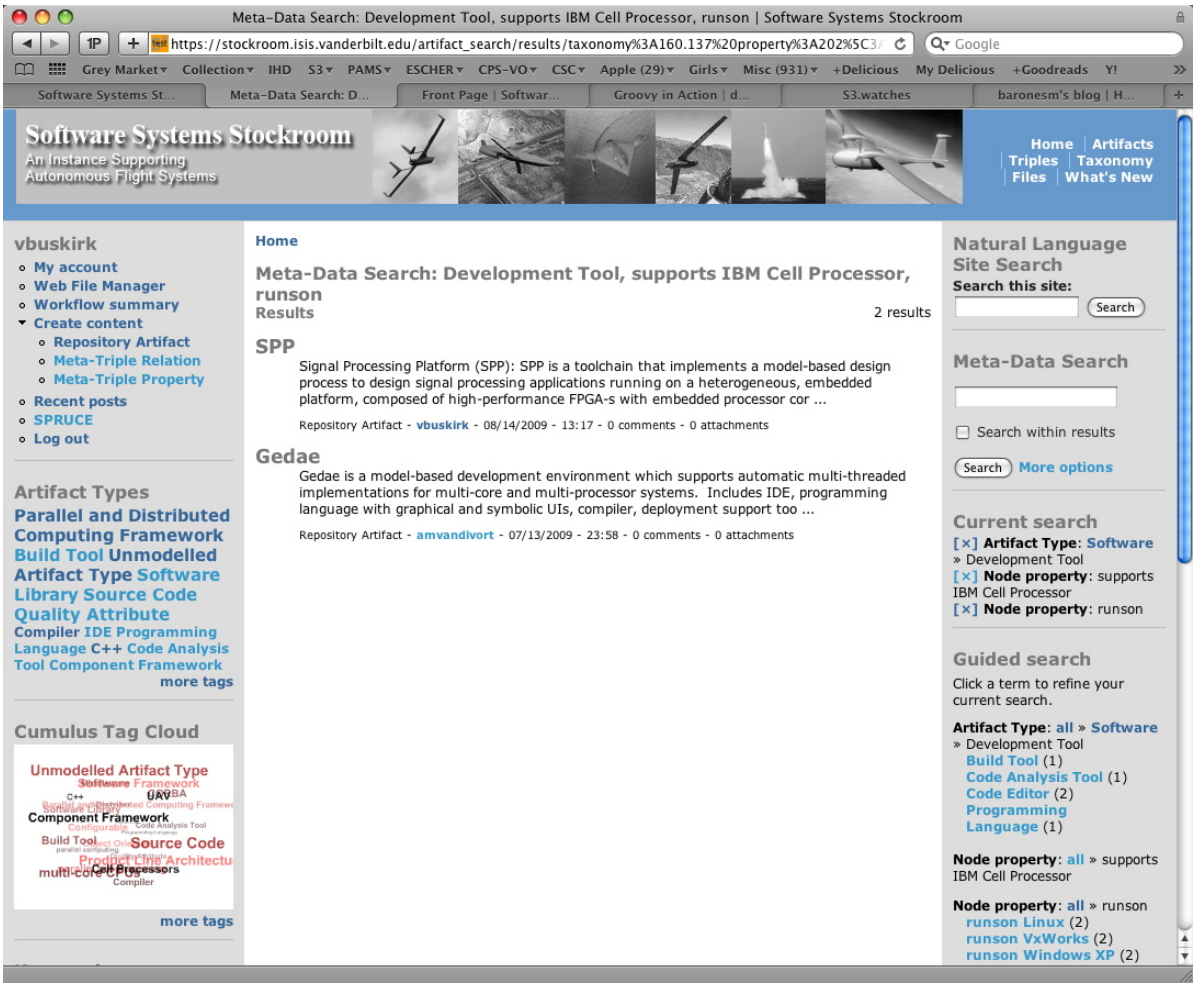


Figure 27: Search Results for Guided Search Involving Properties

Furthermore, we may define expectations about what triples artifacts are expected to have based upon the Stockroom's type system. In the previous two examples, programming tutorials typically have subjects that are programming languages, and integrated development environments *run-on* certain types of development workstations while *generating-code-for* possibly a different set of hardware architecture(s).

Support for triples is implemented with a combination of custom content types, taxonomies and ontologies, PHP scripts, Drupal blocks, and Drupal modules.

Finally, each artifact is associated with zero or more *keywords*. As with the *type* field and the triple *predicates/verbs*, these keywords are associated with a formal Drupal taxonomy. In this case, the user is not required to be a member of the 'admin-taxonomy' role to extend the keywords taxonomy. Keywords may be added on the fly, as needed simply by specifying a comma-delimited list of tokens in the keywords section of the artifact editing form. If a keyword existed previously, the index tables will simply be updated to record that another artifact references this keyword[7]. Otherwise, this new keyword will be automatically appended to the taxonomy.

We see keywords as an important component in the development of more formal ontologies. Throughout the prototyping phase of S3, it was common to discover a keyword that was used repeatedly, or even more likely to recognize a small group of keyword terms that appeared to represent the same concept. By analyzing the keywords taxonomy for the *meaning* of these terms, we frequently identified new concepts that were ultimately migrated from the keywords facility into other, more formal ontologies such as the type system.

4.2.2.2.2.3. Scripts

While the default encoding of a Drupal Node's description section is expected to be HTML, members of the administrative group in Stockroom may define Drupal Nodes that are actually inline PHP code. Figure 28 shows how the report that enumerates triple relations is implemented as PHP code, which traverses the underlying RDBMS tables and filters out rows that fail to meet a certain criteria, finally generating valid HTML on the fly, which displays the results. The links associated with 'Artifacts' and 'Triples' in the upper-right hand navigation menu work similarly at their core by searching through the database tables for the custom content types of Artifacts and Report.

[7] Keywords are searchable.

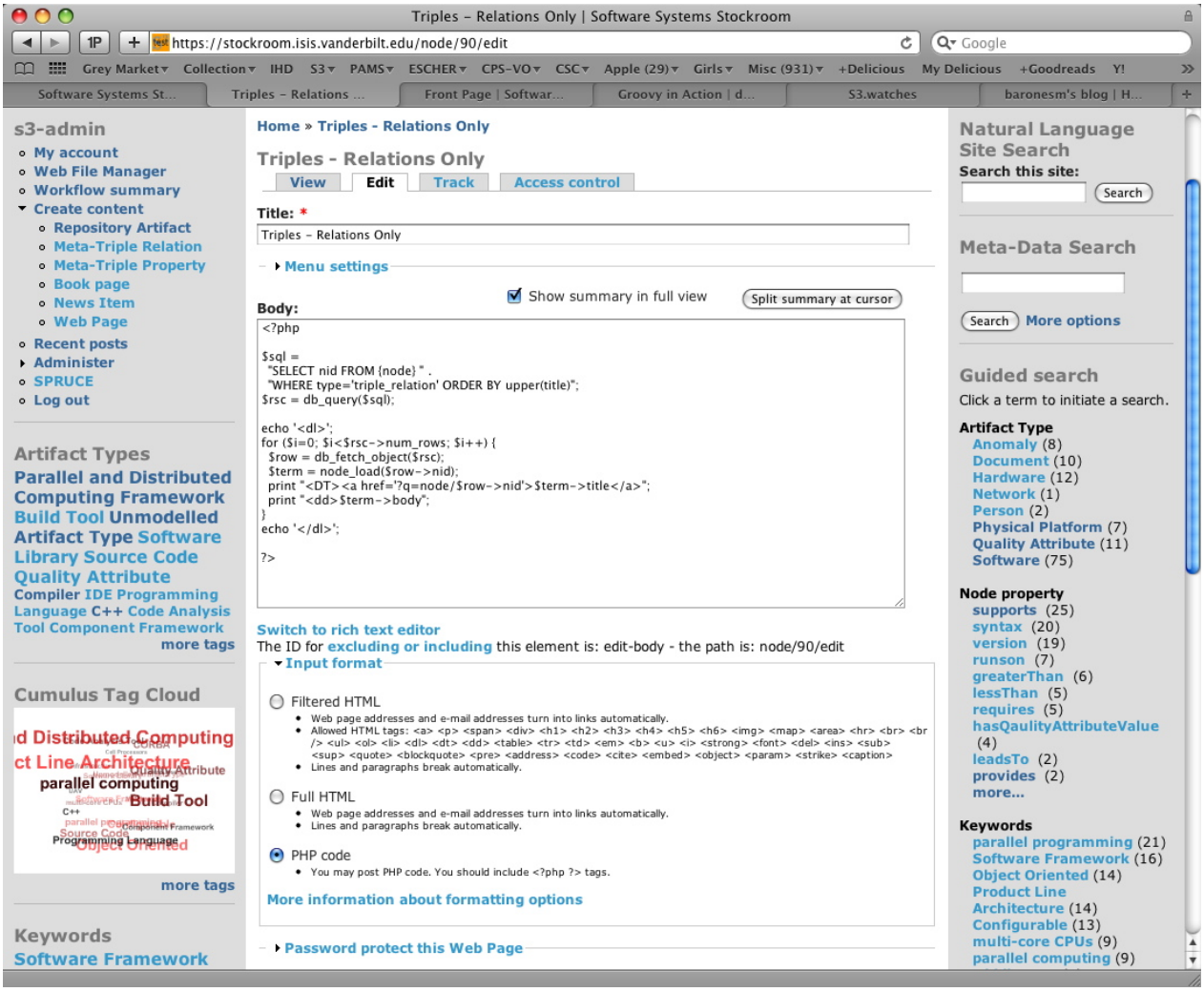


Figure 28: The Triples Report PHP Source

4.2.2.2.3.Policies and Procedures

Any Stockroom artifact may be protected by a user-defined password. Figure 29 shows the *password* section of the artifact editing form.

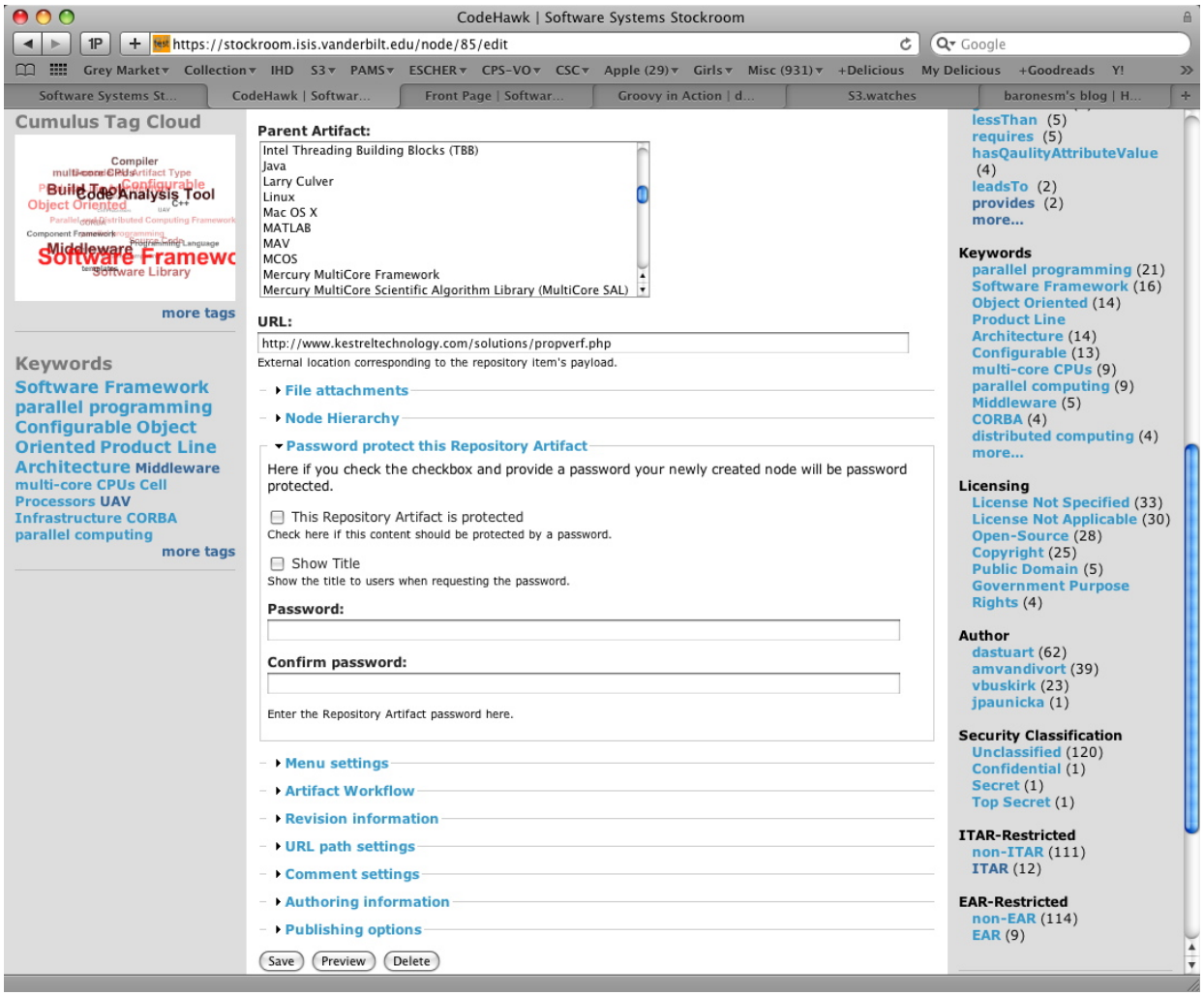


Figure 29: Creating a Password for an Artifact

Additionally content types and individual features may be protected using the notion of roles. In this way, a Stockroom administrator may grant/deny access to all members/non-members of certain groups of users. For example, the following two screenshots show how the WebFM virtual file system is configured to support various working groups. In Figure 30, we see that all TWGs have been defined to have access to the WebFM feature. In Figure 31, we can see the definition of the name of the top-level folder that is allocated for the formal methods TWG.

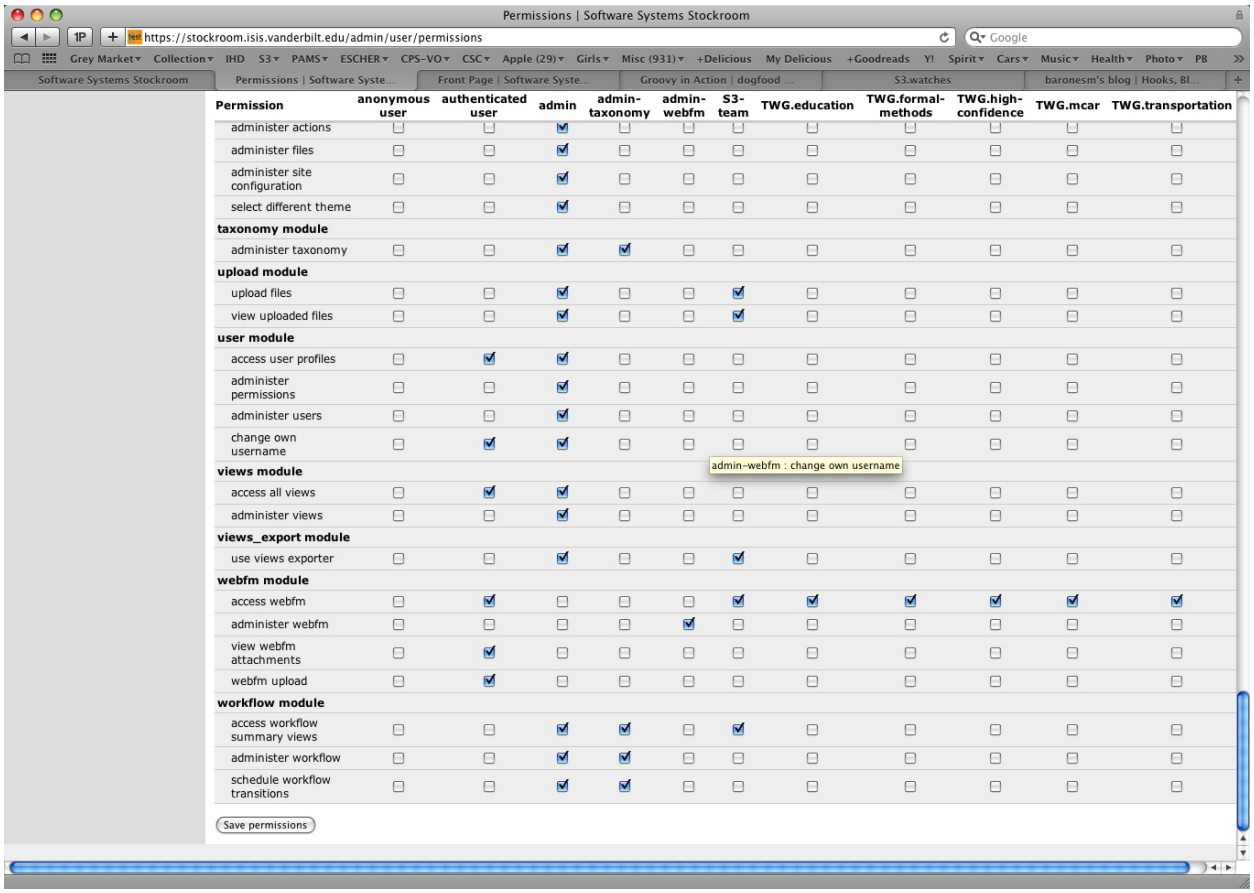


Figure 30: Role Based Access for Working Group Files

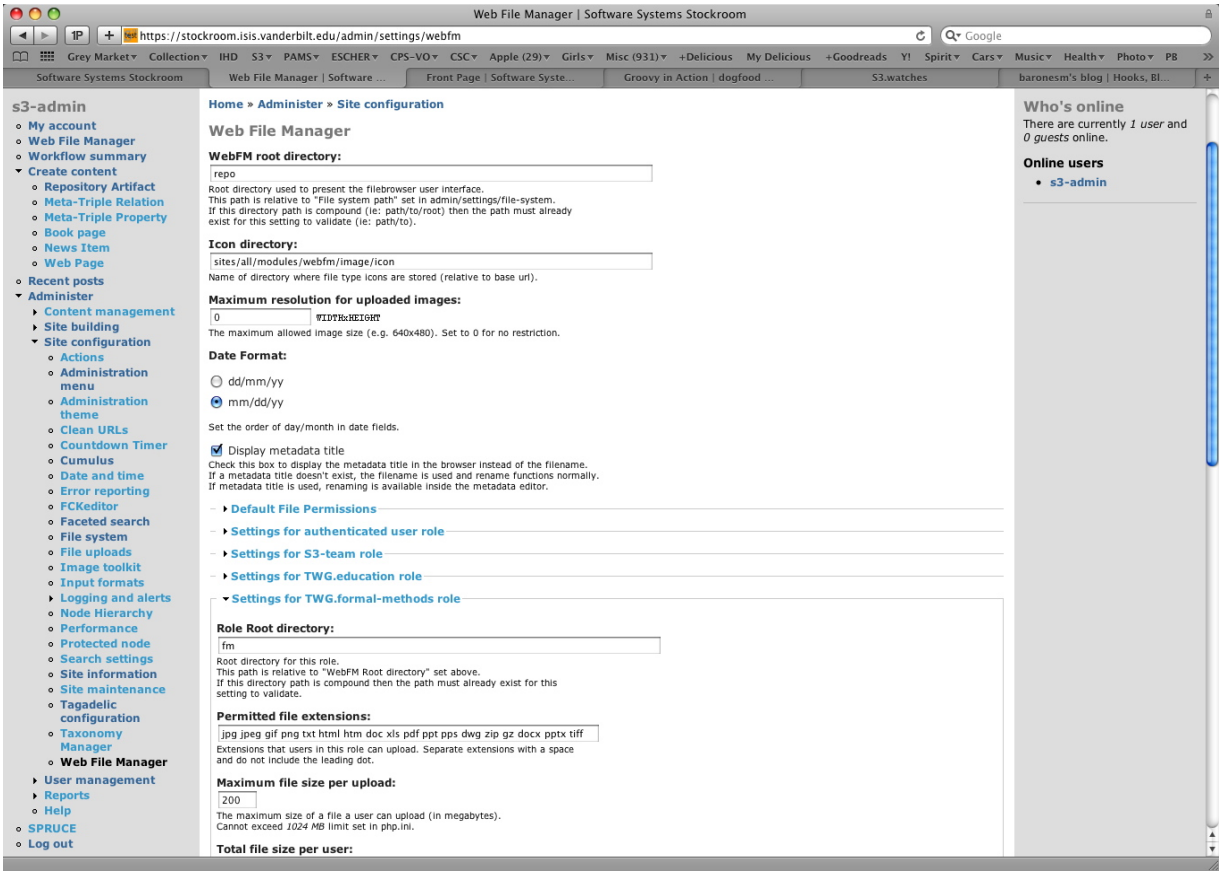


Figure 31: Configuring the Web File Manager for the Formal Methods TWG

When accessing WebFM as a user, one's group membership(s) directly determines the contents visible within WebFM (Figure 32). If I am not a member of the formal methods TWG, the 'fm' directory structure will be unknown to my user account. All WebFM content is visible to the administrator (Figure 33).

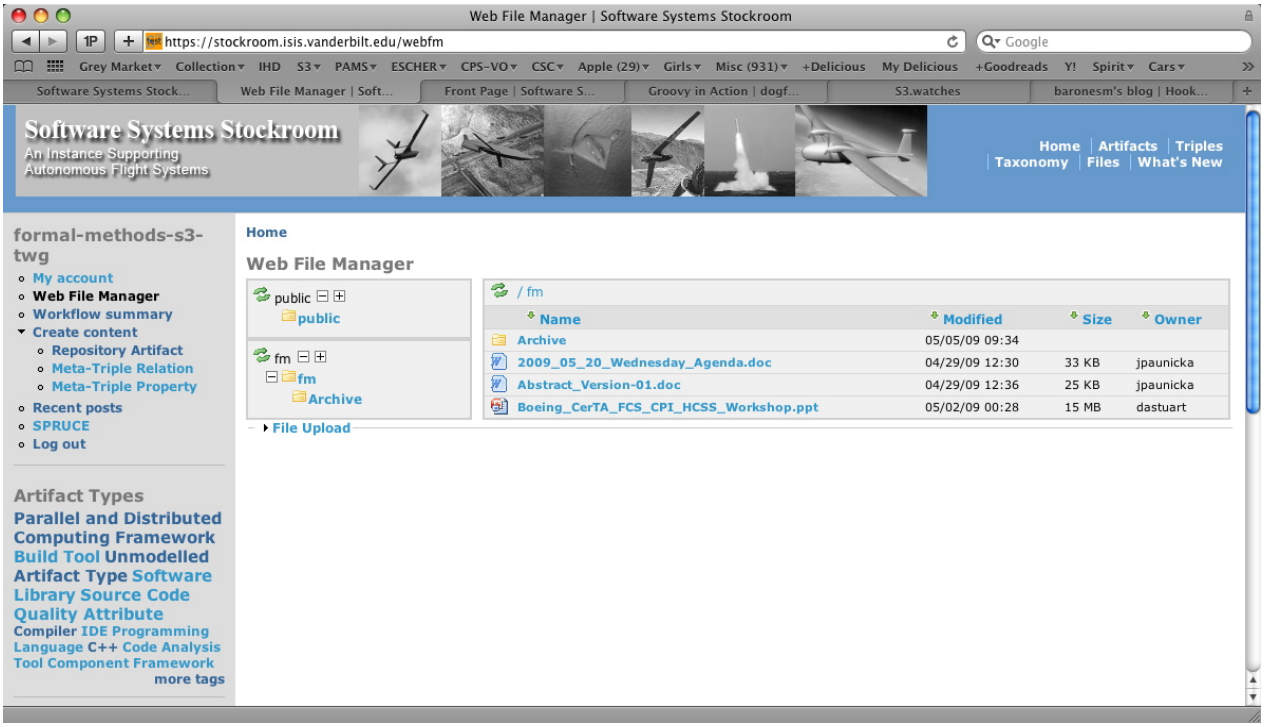


Figure 32: WebFM Portal for Formal Methods TWG Member

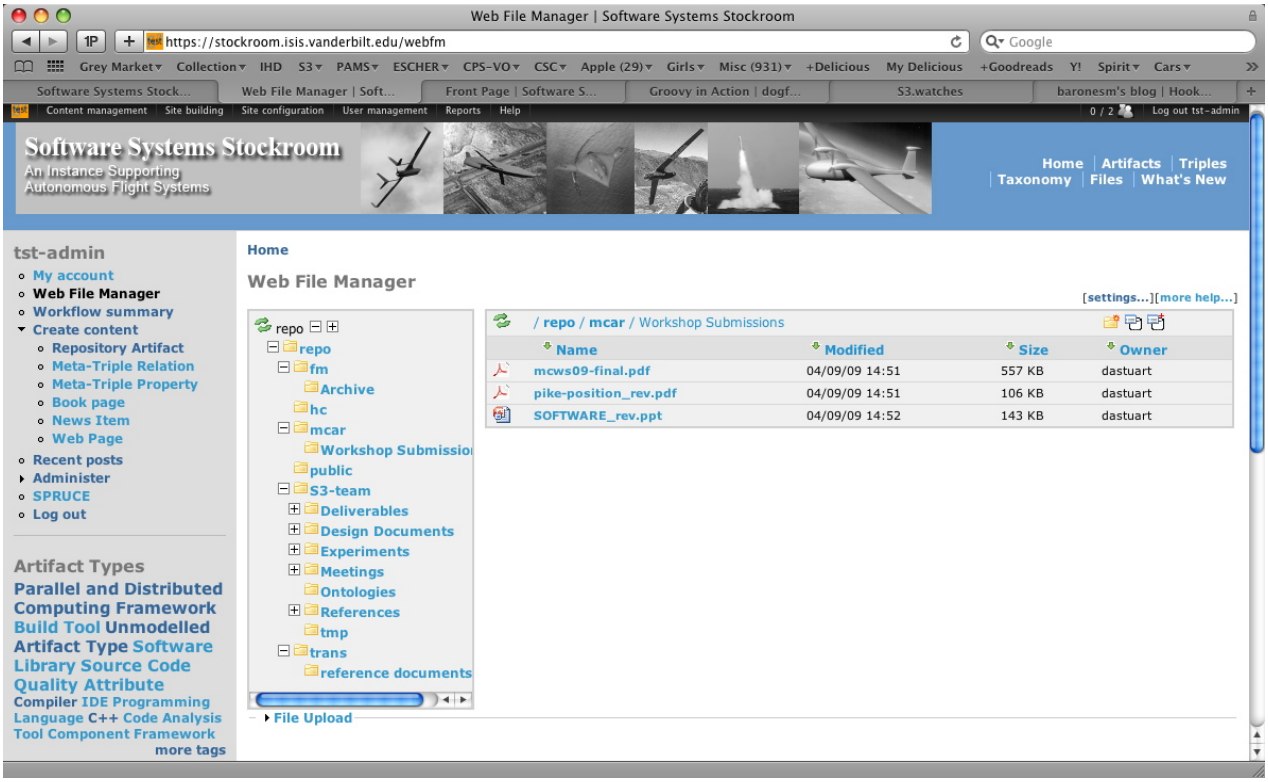


Figure 33: WebFM Portal for Administrative Account

4.2.2.2.3.1. The Workflow Engine

For prototyping purposes, we defined a simple, but extensible workflow (

Figure 34) to be associated with Stockroom artifact contents.

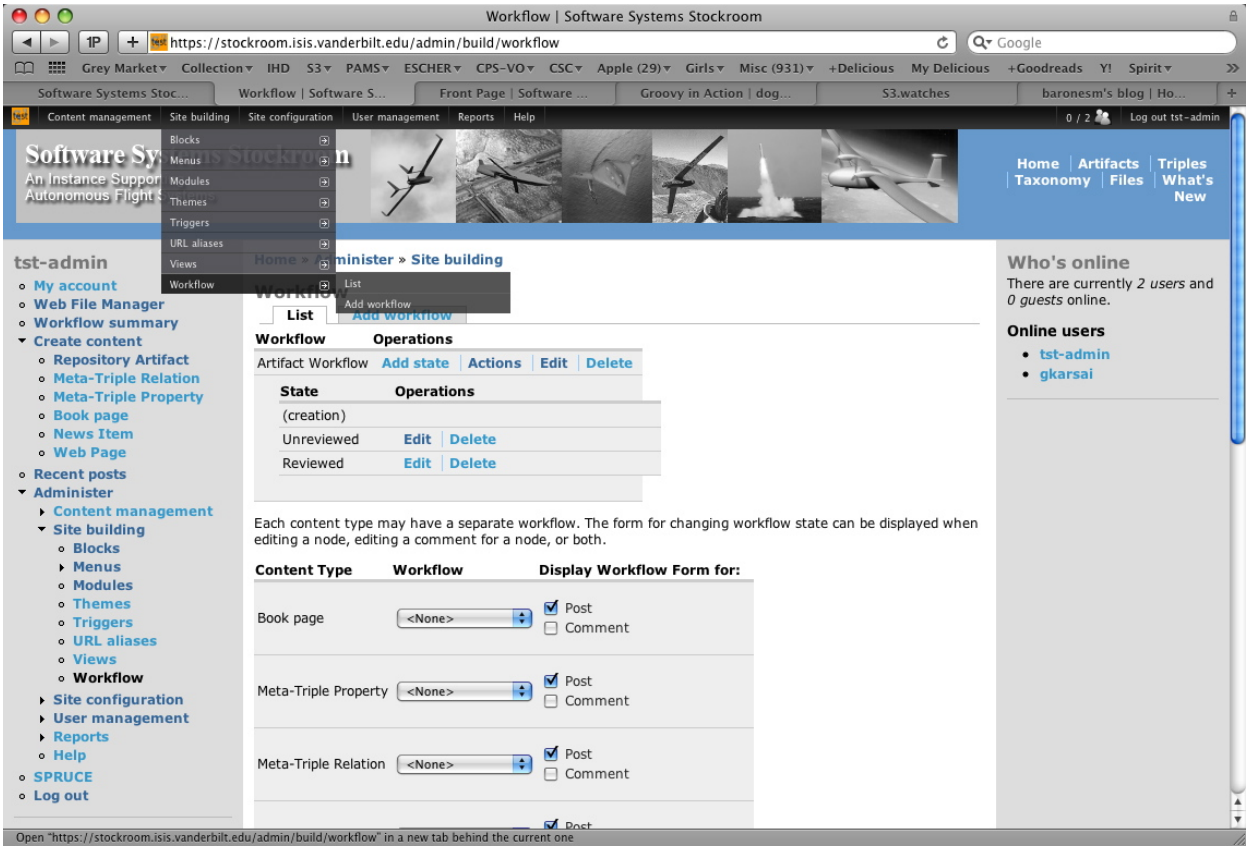


Figure 34: S3 Workflow Engine

In this case, all incoming new Artifacts are initially classified as *Unreviewed*. Additionally, members of the moderators group may transition Artifacts from *Unreviewed* to the *Reviewed* state. Toward this end, that class of users has a "Workflow summary" navigational link within their left sidebar menus.

Figure 35 and

Figure 36 show the workflow in action.

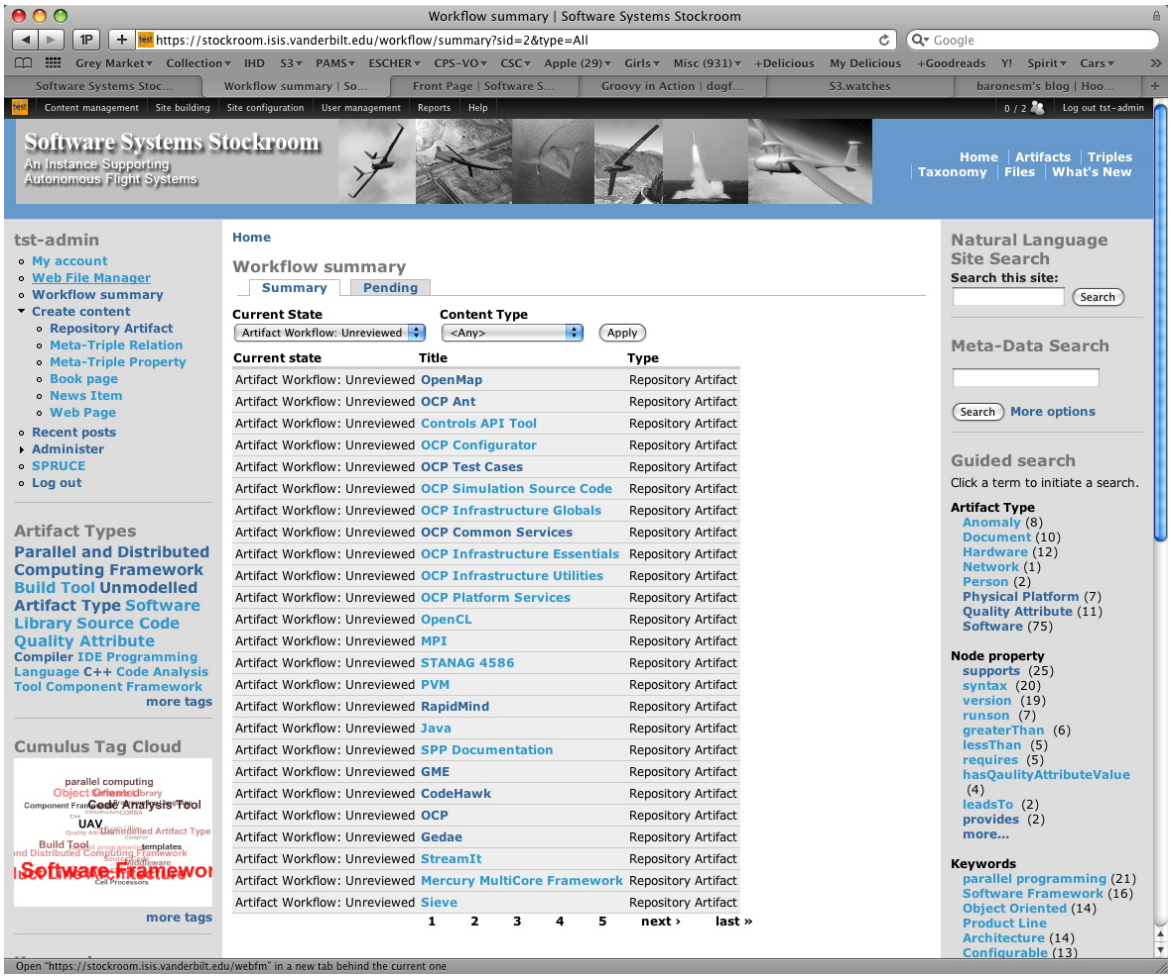


Figure 35: Workflow Summary

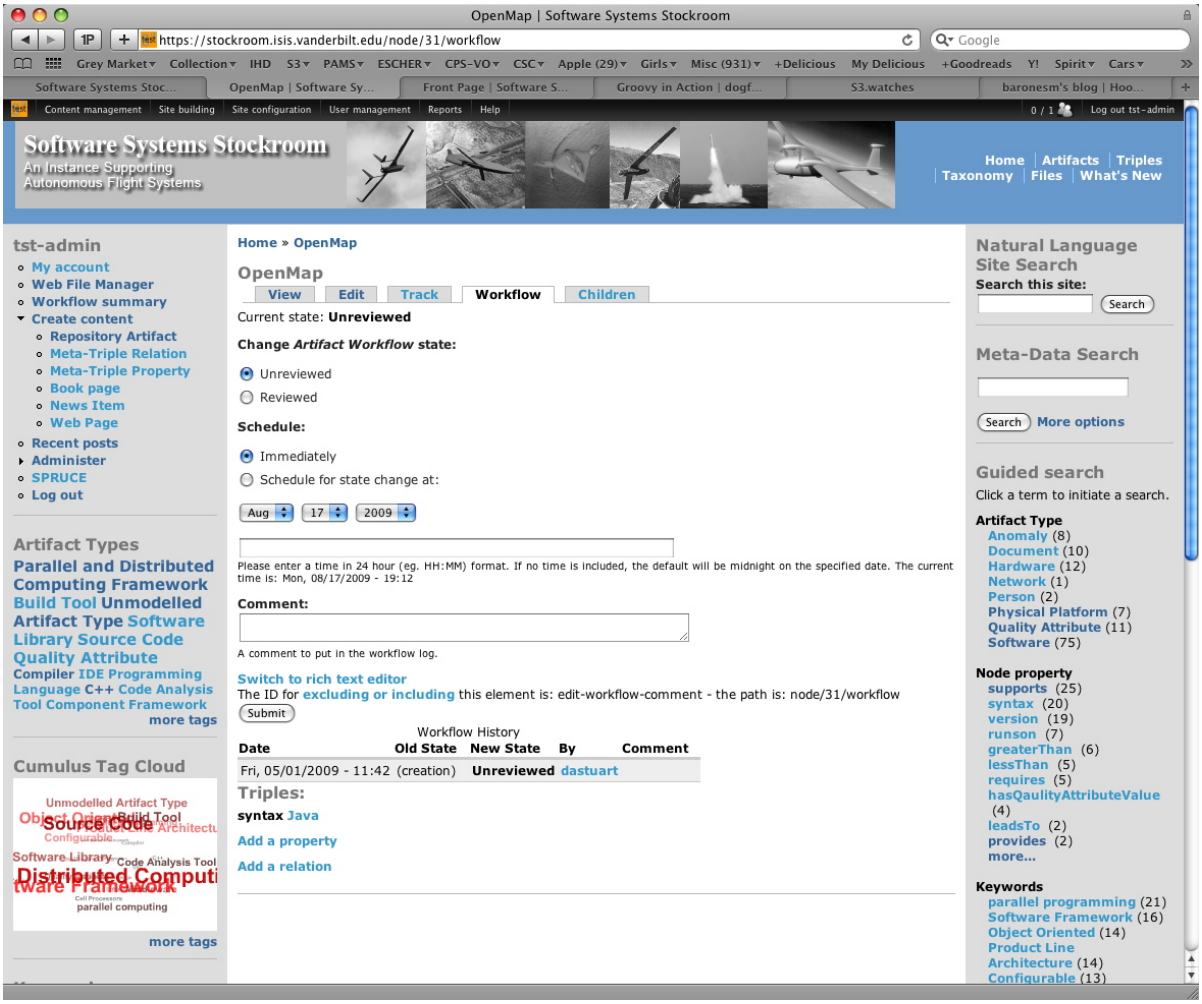


Figure 36: Workflow for an Artifact

4.2.2.2.4.Search

The purpose of the Stockroom is to provide content, and the Stockroom provides three types of search to allow users to identify desired content. Natural language search performs a simple string matching search of the contents of the Stockroom. Metadata search adds additional structure, and provides for search for Stockroom artifacts that are associated with terms from various metadata vocabularies. Finally, ontological search identifies artifacts based on the execution of sophisticated logical queries against a semantic model of the domain.

4.2.2.2.4.1. Natural Language Search

The right-hand sidebars in Stockroom have been configured to display various *search* widgets. The most basic of these is the natural language search. This search is provided out of the box with Drupal, and has not been extended or improved in any significant manner. It is basically a syntactic token matching algorithm that looks for instances of patterns of characters that are found within any section of the Stockroom website. It may be possible to improve upon the quality of the natural language search results by modifying the default implementation of this search engine. However that was out of scope for the prototyping phase of the S3 project[8].

4.2.2.2.4.2. Metadata Search

Stockroom extends the basic syntactic search provided by Drupal with a *faceted search* which has knowledge of the various Stockroom taxonomies. Using this faceted search, we may iteratively refine a search by providing progressively more specific constraints on the desired result set. Consider this example, where we begin the process from the list of all known Stockroom artifacts (Figure 37).

[8] Or by purchasing some 3rd party solution and integrating it into Stockroom using Drupal's module architecture.

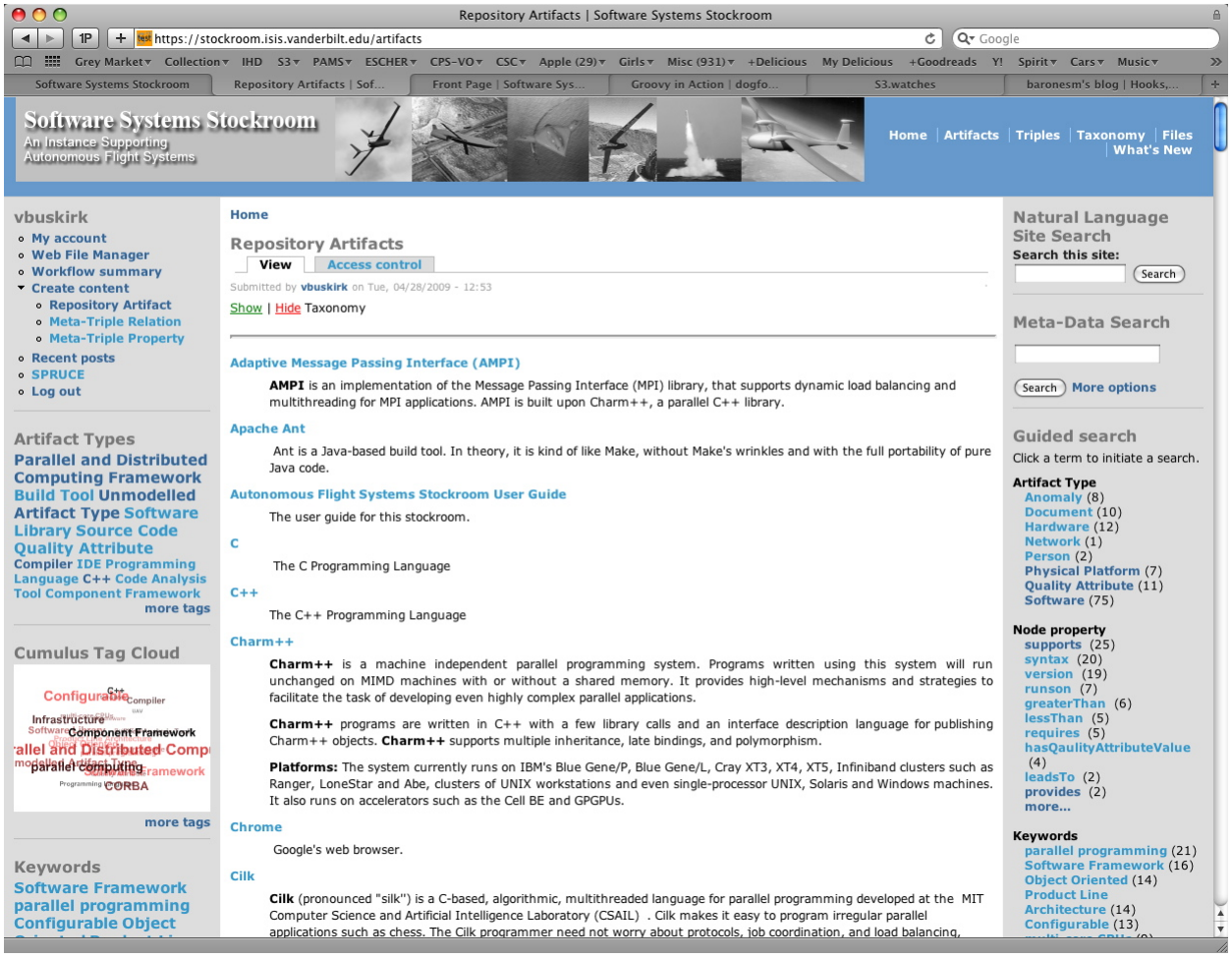


Figure 37: Complete List of Stockroom Artifacts

Next, using the *Guided search* feature in the right-hand sidebar, we specify that we are only interested in software development tools that are classified as Integrated Development Environments (i.e. using the *Artifact Type* facet). In this case, there are now six matches to our query (Figure 38).

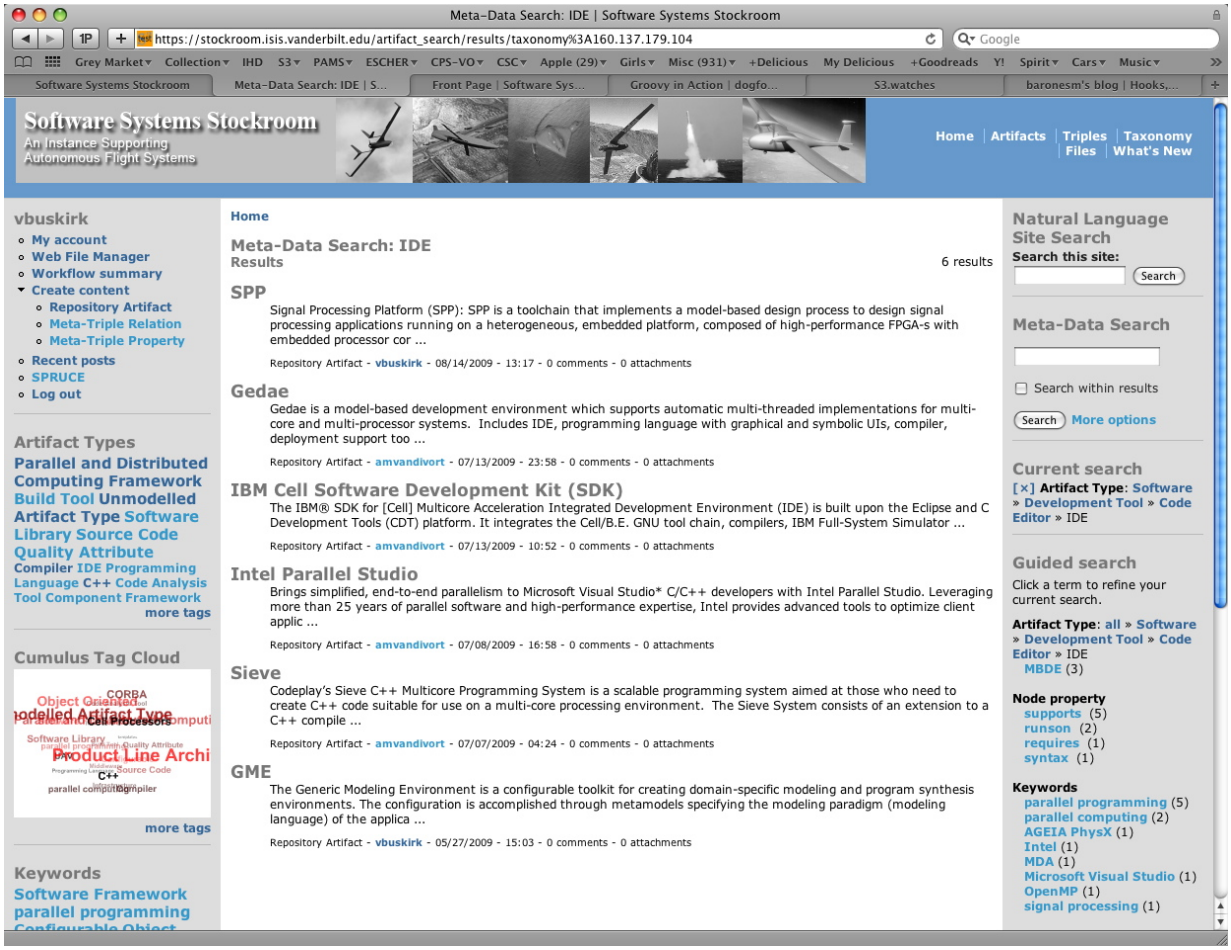


Figure 38: Guided Search Results

If we then refine the search to be restricted to the subset of those Artifacts that were uploaded by someone from Raytheon that also support the IBM Cell Processor, the result set is reduced to three (Figure 39).

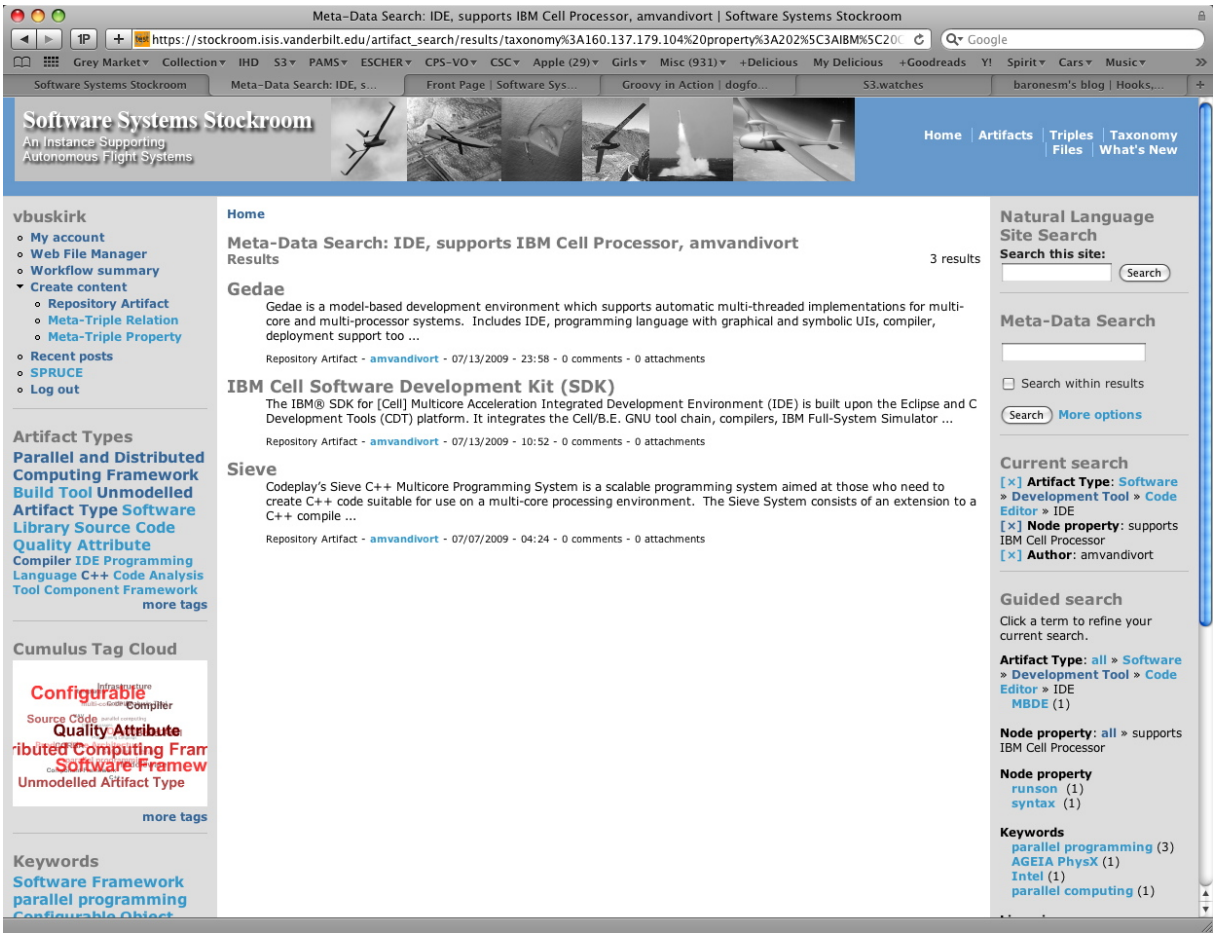


Figure 39: Results of Incremental Search

Finally, if we then back off on the restriction that the items' pedigrees must be associated with Raytheon, then the Stockroom's metadata search engine returns an additional artifact (Figure 40).

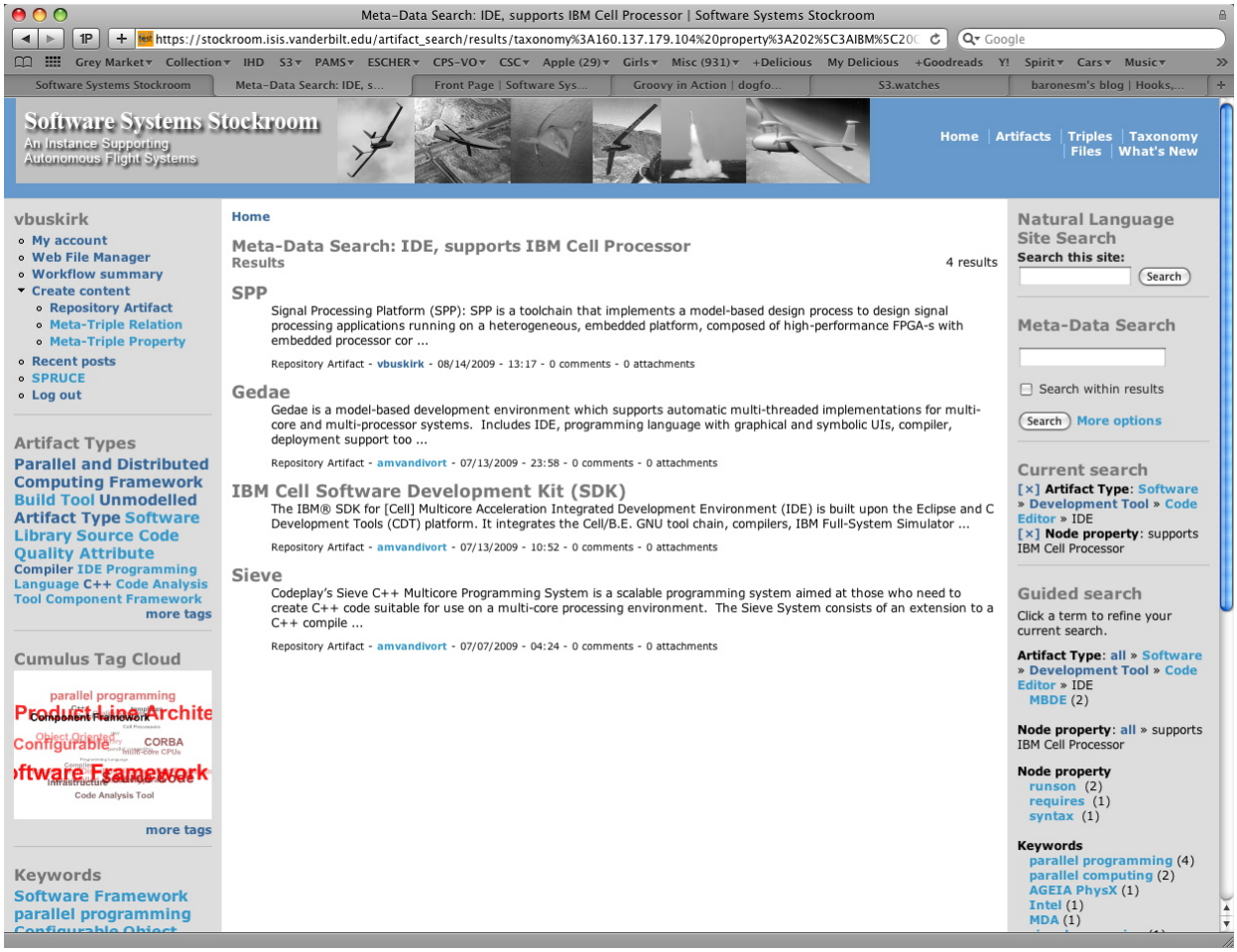


Figure 40: Final Search Results after Unrolling

4.2.2.2.4.3. Ontological Search

The metadata search scenario above is much more powerful than the natural language search described previously in the initial section on search. For example, these metadata searches, which are driven by the underlying taxonomies, can recognize that IDEs are just one of many forms of software development tools (other examples being static analysis tools and programming languages). Also, the natural language search will easily miss items where a user chooses to describe something as an "Integrated Development Environment" rather than the synonymous "IDE" terminology.

Along these same lines, the S3 technology's ontological search further extends the abstraction level, and thus the power, of Stockroom's search capabilities. The following section discusses the ontological search architecture in detail.

At its core, the Drupal content management system stores all data in a relational database management system according to a proprietary database schema. While convenient for prototyping, the drawback of this approach is that the data is not immediately portable to tools that do not support Drupal. To address this issue, these proprietary RDBMS tables, whose layout is determined by the Drupal schemas, were converted to a more standard data-encoding format. As the data housed in S3 is essentially a knowledge base, the formal mathematical logic language chosen was OWL (Web Ontology Language). A background Stockroom daemon executes each minute, and this component performs a simple syntactic translation to convert from RDBMS tables and rows into the OWL format. The following screenshot shows a simple example, where in the OWL language SPP is declared (among other things) to be a model-based development environment that supports the IBM Cell processor (Figure 41).

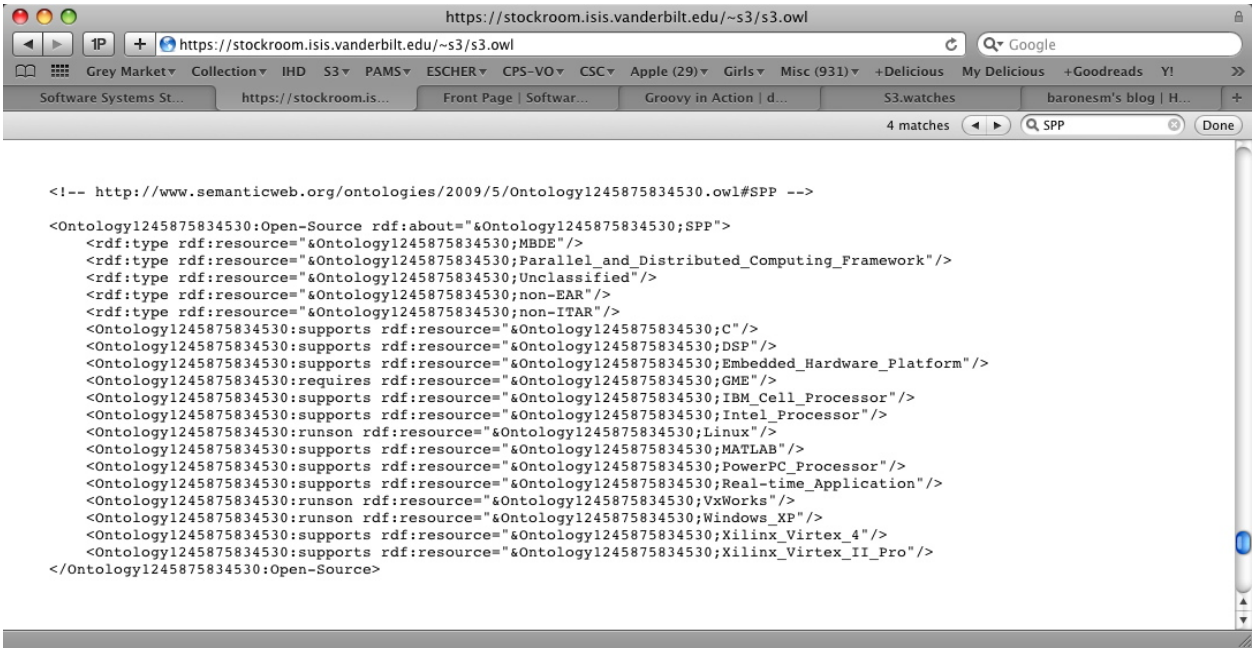


Figure 41: Generating 3.owl

By exporting the Stockroom contents to the OWL syntax, access is gained to a plethora of off-the-shelf tools targeted at this market. While not many 3rd party tools can manipulate the proprietary RDBMS tables of Drupal, there are many different tools that support the OWL language. During various stages of the prototyping phase, for example, we have taken advantage of tools such as CMap (Concept Maps) and Freemind's OWL integration to visualize the contents of the Stockroom database graphically.

Another 3rd party tool we used extensively in conjunction with the S3 project is the Protégé knowledge base editor by Stanford. To make full use of Stockroom, we highly suggest you download and install Protégé on your client machine[9].

[9] http://protege.stanford.edu/

A primary use of Protégé in the Stockroom prototype is to actually implement the Ontological Search feature. As a substrate for our ontological query language, we have chosen a description logic that is defined by the OWL semantic web standard. As Protégé has support for one particular description logic reasoner (viz. FaCT++) with minimal configuration, we again recommend Protégé as the easiest way for a novice user to get started entering and executing description logic queries[10]. The screenshot of Figure 42 shows the results of a Stockroom query for Compilers that support the IBM Cell processor.

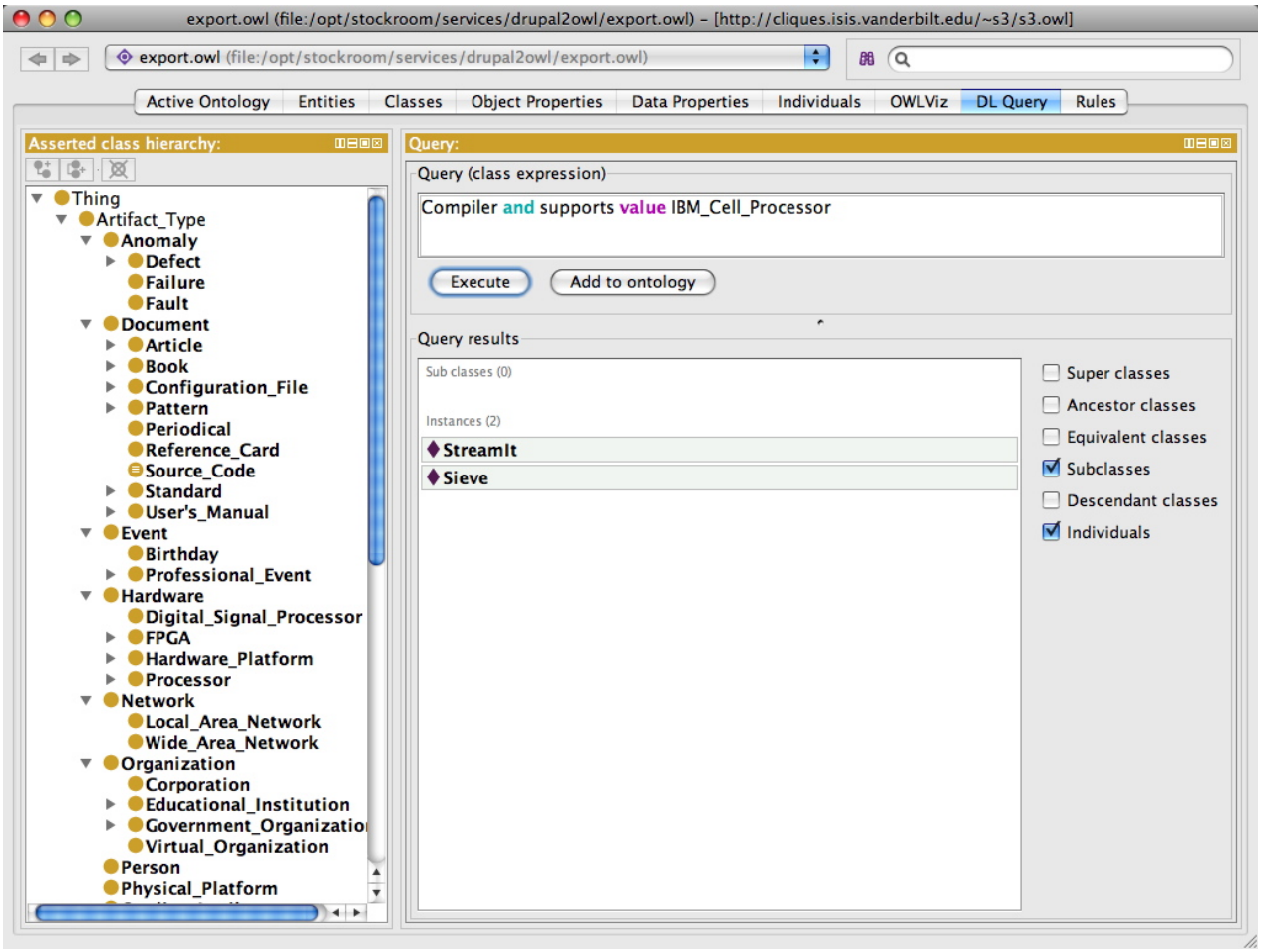


Figure 42: Protégé Query

[10] During the development of the S3 prototype, we also evaluated and had great success with alternatives to the FaCT++ reasoner such as Pellet and Racer.

The semantic web, being based upon open-world reasoning principles, facilitates the aggregation or merging of information from different Stockroom knowledge bases. Due to this open-world property, the various knowledge bases do not have to be complete nor do they need be very tightly coordinated (though this doesn't preclude the possibility). S3 therefore is somewhat forgiving about different Stockrooms' slightly varying views or models of the world. So long as no facts declared at one site are in outright conflict with any facts stated in a second site, the Stockroom reasoners are quite satisfied. In the event that a conflict does exist, this anomaly is easily detected programmatically by the ontological reasoners, and Stockroom administrators are notified of the conflict.

With this in mind, we can see that multiple stockrooms may be easily federated simply by combining their logic-based representations into a single, aggregate store. Furthermore, by declaring and following some simple rules about which sites may or may not access other sites, we can easily enforce some information hiding principles to create stratums of federations. Using this strategy, the Drupal-To-Owl component combines and merges OWL declarations from different sites to create aggregate knowledge bases.

4.2.2.3. Metrics Reporting

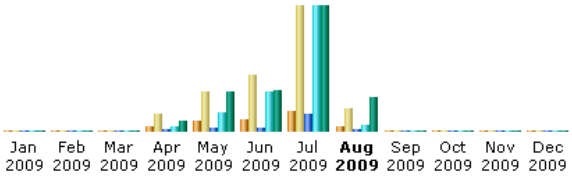
The Stockroom leverages *AWStats*, which is an open source tool for analyzing web server logs. We provide reports that include a number of useful metrics. We track metrics such as the number of visits, number of unique visitors, and web page hits over time. The generated reports display graphs of these metrics on an hourly, daily (Figure 43), and monthly (Figure 44) basis, making it easy to see how the website grows over time.

Days of month

Day	Number of visits	Pages	Hits	Bandwidth
01 Aug 2009	0	0	32	294.88 KB
02 Aug 2009	1	13	77	589.75 KB
03 Aug 2009	0	0	176	1.58 MB
04 Aug 2009	9	145	537	4.98 MB
05 Aug 2009	4	48	1117	3.52 MB
06 Aug 2009	12	595	1945	13.47 MB
07 Aug 2009	6	213	511	109.47 MB
08 Aug 2009	0	0	32	294.88 KB
09 Aug 2009	1	30	62	294.88 KB
10 Aug 2009	7	58	422	33.48 MB
11 Aug 2009	1	1	162	1.45 MB
12 Aug 2009	10	50	431	5.46 MB
13 Aug 2009	14	198	677	18.48 MB
14 Aug 2009	0	0	0	0
15 Aug 2009	0	0	0	0
16 Aug 2009	0	0	0	0
17 Aug 2009	0	0	0	0
18 Aug 2009	0	0	0	0
19 Aug 2009	0	0	0	0
20 Aug 2009	0	0	0	0
21 Aug 2009	0	0	0	0
22 Aug 2009	0	0	0	0
23 Aug 2009	0	0	0	0
24 Aug 2009	0	0	0	0
25 Aug 2009	0	0	0	0
26 Aug 2009	0	0	0	0
27 Aug 2009	0	0	0	0
28 Aug 2009	0	0	0	0
29 Aug 2009	0	0	0	0
30 Aug 2009	0	0	0	0
31 Aug 2009	0	0	0	0
Average	5.00	103.92	475.46	14.87 MB
Total	65	1351	6181	193.32 MB

Figure 43: Daily Statistics

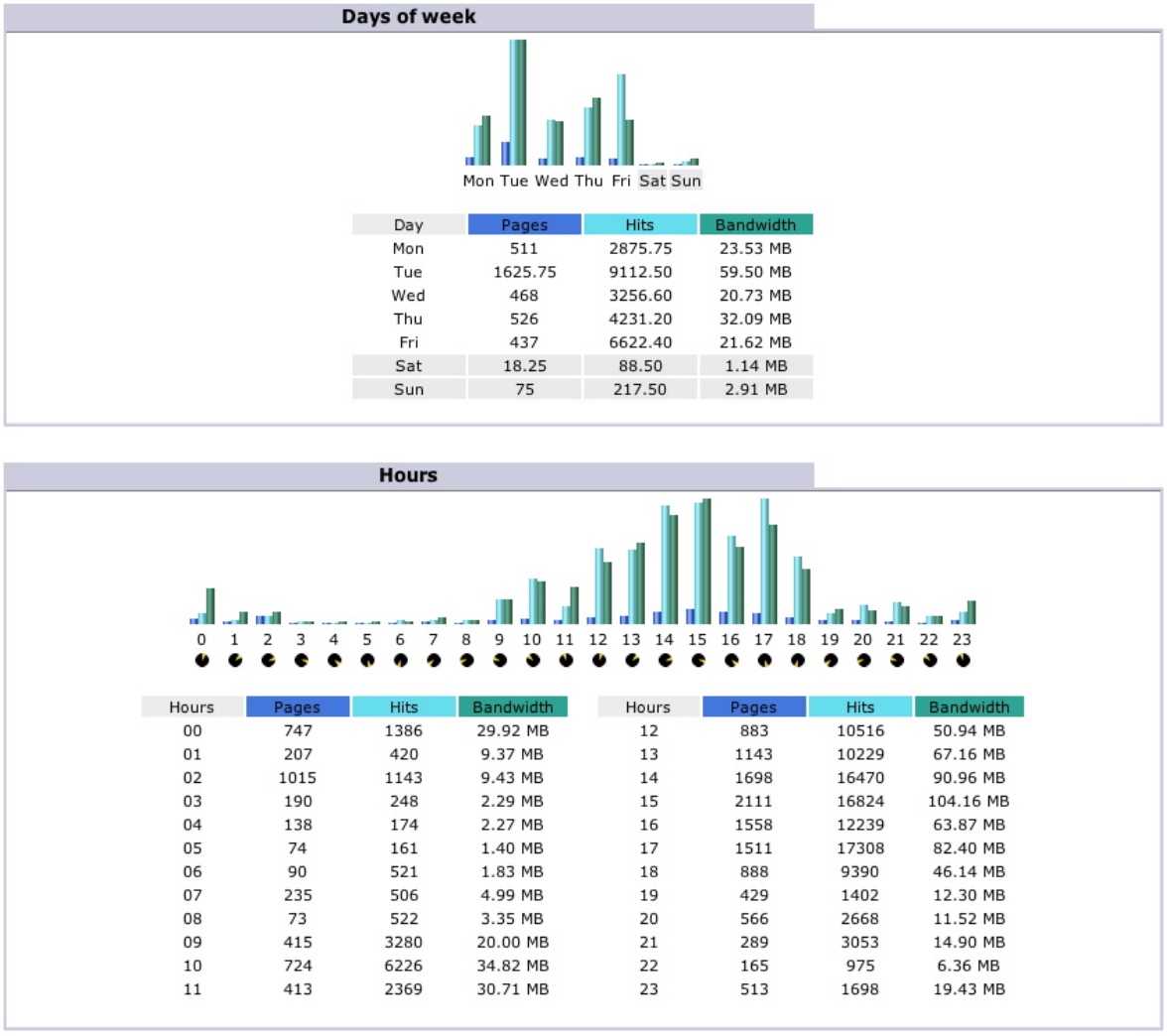
Monthly history



Month	Unique visitors	Number of visits	Pages	Hits	Bandwidth
Jan 2009	0	0	0	0	0
Feb 2009	0	0	0	0	0
Mar 2009	1	2	30	30	0
Apr 2009	15	48	1581	4014	61.20 MB
May 2009	28	111	3713	18399	227.14 MB
Jun 2009	33	156	2823	38376	236.90 MB
Jul 2009	56	351	16075	119728	720.53 MB
Aug 2009	15	65	1351	6181	193.32 MB
Sep 2009	0	0	0	0	0
Oct 2009	0	0	0	0	0
Nov 2009	0	0	0	0	0
Dec 2009	0	0	0	0	0
Total	148	733	25573	186728	1.41 GB

Figure 44: Monthly Statistics

The following report (Figure 45) shows that nearly all visits came during the workweek, and that these visits happened mostly during normal business hours.



Day	Pages	Hits	Bandwidth
Mon	511	2875.75	23.53 MB
Tue	1625.75	9112.50	59.50 MB
Wed	468	3256.60	20.73 MB
Thu	526	4231.20	32.09 MB
Fri	437	6622.40	21.62 MB
Sat	18.25	88.50	1.14 MB
Sun	75	217.50	2.91 MB

Hours	Pages	Hits	Bandwidth	Hours	Pages	Hits	Bandwidth
00	747	1386	29.92 MB	12	883	10516	50.94 MB
01	207	420	9.37 MB	13	1143	10229	67.16 MB
02	1015	1143	9.43 MB	14	1698	16470	90.96 MB
03	190	248	2.29 MB	15	2111	16824	104.16 MB
04	138	174	2.27 MB	16	1558	12239	63.87 MB
05	74	161	1.40 MB	17	1511	17308	82.40 MB
06	90	521	1.83 MB	18	888	9390	46.14 MB
07	235	506	4.99 MB	19	429	1402	12.30 MB
08	73	522	3.35 MB	20	566	2668	11.52 MB
09	415	3280	20.00 MB	21	289	3053	14.90 MB
10	724	6226	34.82 MB	22	165	975	6.36 MB
11	413	2369	30.71 MB	23	513	1698	19.43 MB

Figure 45: Comparing Traffic by Day of Week and Time of Day

From the next screenshots (Figure 46), one can glean that the primary customers of the facility are from Vanderbilt, Boeing and Raytheon.

Visitors domains/countries (Top 10) - Full list

Domains/Countries		Pages	Hits	Bandwidth
USA Educational	edu	5807	79766	368.31 MB
Commercial	com	5328	11642	201.28 MB
Unknown	ip	4593	23958	118.60 MB
Network	net	335	4350	31.58 MB
United Kingdom	uk	12	12	789.25 KB
Others		0	0	0

Hosts (Top 10) - Full list - Last visit - Unresolved IP Address

Hosts : 45 Known, 15 Unknown (unresolved ip) 56 Unique visitors	Pages	Hits	Bandwidth	Last visit
dhcp-129-59-129-199.n1.vanderbilt.edu	4503	74287	273.20 MB	30 Jul 2009 - 17:34
stl-proxy-08.boeing.com	1288	2370	59.19 MB	28 Jul 2009 - 17:47
tus-gate3.raytheon.com	1249	2209	32.42 MB	15 Jul 2009 - 12:29
localhost	1242	1242	0	31 Jul 2009 - 20:23
dhcp-129-59-129-171.n1.vanderbilt.edu	1225	4944	79.20 MB	30 Jul 2009 - 13:49
130.76.145.155	1209	16374	62.76 MB	15 Jul 2009 - 11:52
tus-gate1.raytheon.com	1195	1971	36.52 MB	23 Jul 2009 - 15:38
65.175.116.95	986	986	7.69 MB	14 Jul 2009 - 02:49
tus-gate2.raytheon.com	435	999	14.44 MB	15 Jul 2009 - 12:29
130.76.32.181	384	743	19.09 MB	31 Jul 2009 - 20:24
Others	2359	13603	136.02 MB	

Figure 46: Traffic Source Statistics

The policy that non-authenticated users are not allowed to access Stockroom data is correctly being enforced (Figure 47, Figure 48).

Robots/Spiders visitors (Top 10) - Full list - Last visit

0 different robots*	Hits	Bandwidth	Last visit

* Robots shown here gave hits or traffic "not viewed" by visitors, so they are not included in other charts.

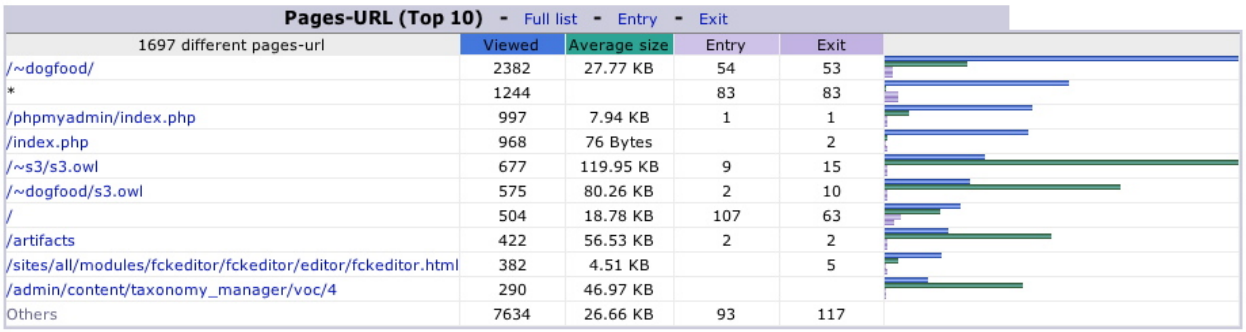
Figure 47: Spider Visits

HTTP Status codes

	HTTP Status codes*	Hits	Percent	Bandwidth
404	Document Not Found	6713	66.1 %	8.72 MB
302	Moved temporarily (redirect)	2524	24.8 %	0
403	Forbidden	768	7.5 %	2.70 MB
400	Bad Request	50	0.4 %	18.98 KB
405	Method not allowed	33	0.3 %	13.05 KB
301	Moved permanently (redirect)	20	0.1 %	8.01 KB
503	Server busy	19	0.1 %	56.59 KB
401	Unauthorized	18	0.1 %	9.78 KB
206	Partial Content	4	0 %	5.10 MB
417	Failed	3	0 %	1.63 KB

Figure 48: HTTP Status Codes

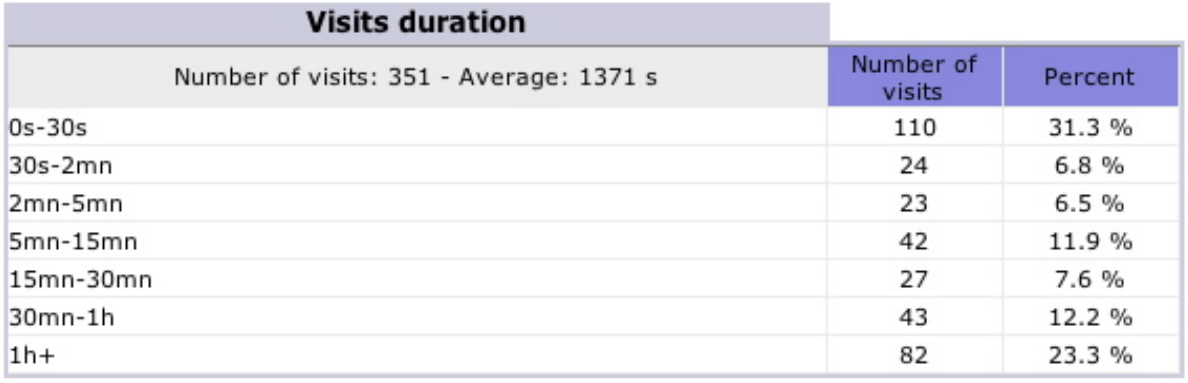
The reports also showed which pages were the most visited (Figure 49), which is useful for determining how the website is used and which features are most popular. AWStats also computes the lengths of every visit for a given month (Figure 50). For example, 10% of visits for July lasted between 5 minutes and 15 minutes. As the website's content grows, the average visit's duration may become longer.



Pages-URL (Top 10) - Full list - Entry - Exit

1697 different pages-url	Viewed	Average size	Entry	Exit
/~dogfood/	2382	27.77 KB	54	53
*	1244		83	83
/phpmyadmin/index.php	997	7.94 KB	1	1
/index.php	968	76 Bytes		2
/~s3/s3.owl	677	119.95 KB	9	15
/~dogfood/s3.owl	575	80.26 KB	2	10
/	504	18.78 KB	107	63
/artifacts	422	56.53 KB	2	2
/sites/all/modules/fckeditor/fckeditor/editor/fckeditor.html	382	4.51 KB		5
/admin/content/taxonomy_manager/voc/4	290	46.97 KB		
Others	7634	26.66 KB	93	117

Figure 49: Frequently Viewed Pages



Visits duration

Number of visits: 351 - Average: 1371 s	Number of visits	Percent
0s-30s	110	31.3 %
30s-2mn	24	6.8 %
2mn-5mn	23	6.5 %
5mn-15mn	42	11.9 %
15mn-30mn	27	7.6 %
30mn-1h	43	12.2 %
1h+	82	23.3 %

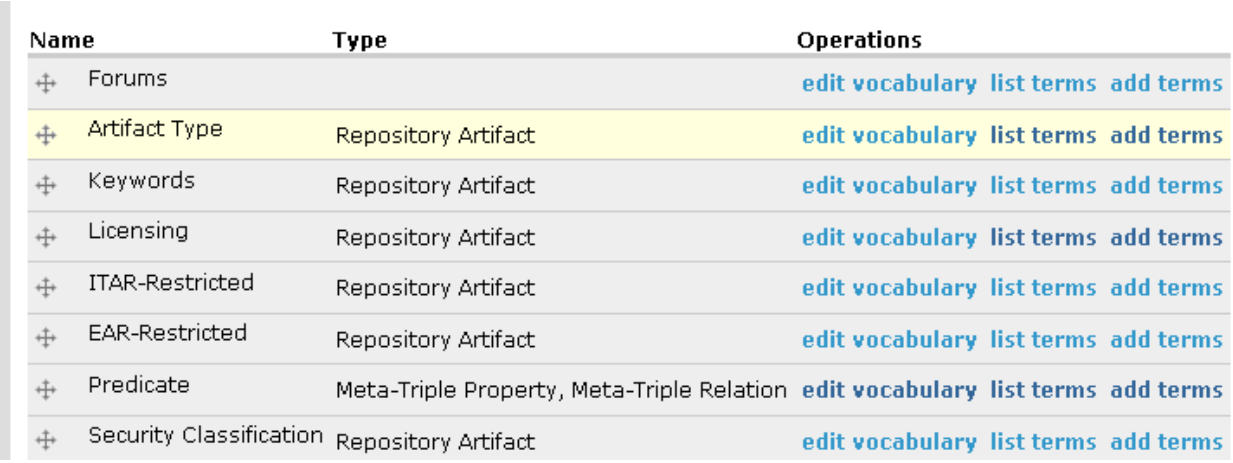
Figure 50: Dwell Time

4.2.3. Domain Taxonomy and Ontology

To enable the metadata and ontology search capabilities provided by the stockroom, metadata describing the contents of the stockroom in domain specific terms of reference must be associated with the stockroom artifacts. The stockroom uses taxonomies to capture some of this metadata. More general metadata is captured in an ontology. Since both the domain and the stockroom are constantly evolving, the approach to both the taxonomies and the ontology has been chosen with an eye to supporting their evolution.

4.2.3.1. Taxonomies

The stockroom taxonomies represent a constrained subset of an artifacts metadata, and generally represent attributes that apply to all or almost all of the artifacts. Taxonomies are generally represented natively in Drupal, with additional supplemental tools developed to provide additional support for taxonomy evolution. Figure 51 shows the taxonomies implemented in the prototype stockroom. Six of the taxonomies apply to stockroom artifacts: Artifact Type, Keywords, Licensing, ITAR-Restricted, EAR-Restricted and Security Classification. Each artifact has associated with it a single value from each of the Licensing, ITAR-Restricted, EAR-Restricted and Security Classification taxonomies. Each artifact can have one or more associated values from the Artifact Type and Keywords taxonomies.



Name	Type	Operations
Forums		edit vocabulary list terms add terms
Artifact Type	Repository Artifact	edit vocabulary list terms add terms
Keywords	Repository Artifact	edit vocabulary list terms add terms
Licensing	Repository Artifact	edit vocabulary list terms add terms
ITAR-Restricted	Repository Artifact	edit vocabulary list terms add terms
EAR-Restricted	Repository Artifact	edit vocabulary list terms add terms
Predicate	Meta-Triple Property, Meta-Triple Relation	edit vocabulary list terms add terms
Security Classification	Repository Artifact	edit vocabulary list terms add terms

Figure 51: Stockroom Taxonomies

The ITAR-Restricted and EAR-Restricted taxonomies represent the possible states of an artifact with respect to the ITAR and EAR export control regimes, and for the purposes of the prototype stockroom, the two values correspond to restricted and non-restricted artifacts. Similarly, the Security Classification taxonomy characterizes artifacts as unclassified, confidential, secret or top secret.

The Licensing taxonomy, though still associating only a single taxonomy value to an artifact, has additional structure. There are open source and non-open source licenses. Many open source licenses have gone through multiple versions. These relationships can be shown in a taxonomy by means of the taxonomy hierarchy. These relationships can be the subject of queries. For example, a search for a tool subject to an open source license would be satisfied by any artifact with any of the specific licenses that are represented by license taxonomy terms that are descendants of the open source taxonomy term. Figure 52 shows a subset of the licensing taxonomy displayed using the tool Protégé, one of the tools used in developing the stockroom taxonomies and ontology. The Licensing Taxonomy is one of the taxonomies that will clearly evolve with the stockroom. As artifacts are added to the taxonomy, their licenses may need to be added to the taxonomy, and new licenses are continually being created.

Figure 52: Subset of the Licensing Taxonomy in Protégé

The Artifact Type taxonomy provides basic classification information (a taxonomy) for stockroom artifacts. Essentially the Artifact Type is used to capture the "IsA" property of stockroom artifacts. Since artifacts can be of multiple types, more than one term from the Artifact Taxonomy may be associated with each artifact. As with the Licensing taxonomy, the Artifact Type taxonomy also has a hierarchical structure. Figure 53 shows an extract from the Artifact Type taxonomy as displayed in Drupal.

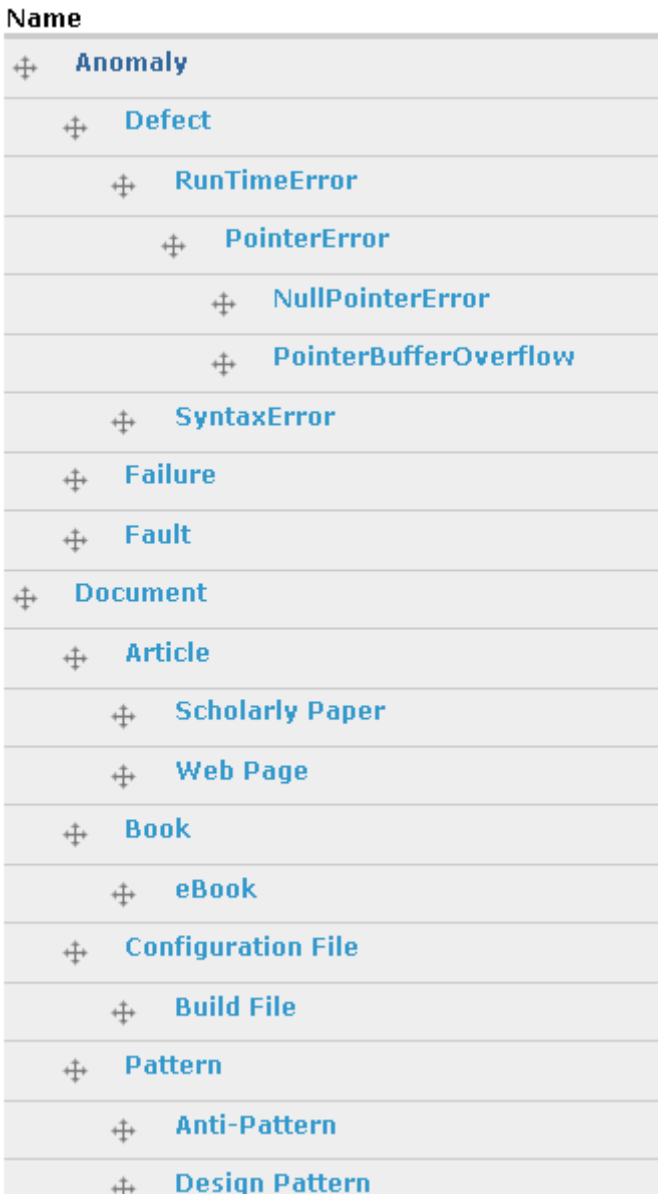


Figure 53: Artifact Type Taxonomy Abstract in Drupal

The final artifact taxonomy in the prototype stockroom is the keywords taxonomy. The keywords taxonomy is, as the name implies, a way to associate one or more keywords with a stockroom artifact. The keyword taxonomy is used to define those attributes of an artifact that are not covered by the other taxonomies. Unlike the other taxonomies, any user can propose a new keyword simply by supplying it as a value of the keyword attribute when creating or editing the metadata of an artifact. This ability of any user to propose a keyword is a manifestation of the dynamic nature of the stockroom domain. This ensures that the structure of the stockroom does not inhibit capture of domain knowledge.

4.2.3.2. Ontology

The stockroom taxonomies support more precise retrieval of stockroom artifacts by associating identified terms with artifacts in such a way that artifacts explicitly associated with those terms can be found. This is an improvement over natural language search in that an artifact only matches the term if the term, in its specialized taxonomic use, has been declared to apply to the artifact. Though more precise than natural language search, such a search is still limited in power and scope. To fully realize the potential of the stockroom, more powerful techniques are required, including the ability to construct queries that reason over the available information to enable still more precise identification of stockroom artifacts. The stockroom uses an ontology, and an ontological search, to achieve this higher level of precision. Figure 54 is representative of the type of query the stockroom should ultimately be able to support, and the ontological information that would enable such a query. The prototype stockroom, and its ontology, was developed to support such queries.

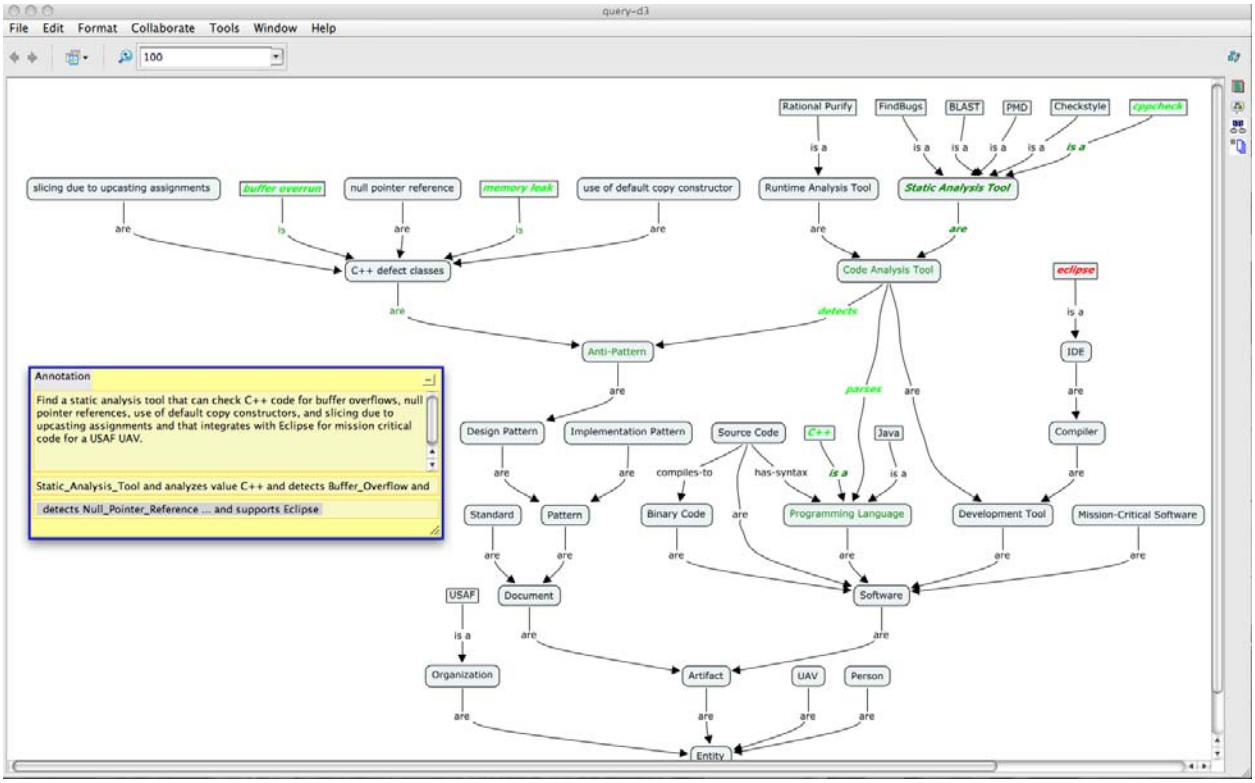


Figure 54: Representative Ontological Query

The stockroom ontology is represented using a combination of RDF style triples and an OWL ontology. The primary ontological metadata, in the form of triples, captures relationships between an artifact and either another artifact (relations) or a taxonomy term (properties). Individual facts (instances of properties or relations) are stored and manipulated as triples in the stockroom.

Figure 55 is a subset of the set of all triples stored in the stockroom as displayed when selecting the Triples shortcut menu option. The triples are displayed in prefix relation notation, where *Foo(bar, baz)* means that artifact *bar* stands in relation *foo* to *baz*, where *baz* is either a taxonomy term or another artifact. As an example from

Figure 55, *detects(CodeHawk, PointerBufferOverflow)* states that the artifact *CodeHawk detects PointerBufferOverflow*, where CodeHawk is a formal methods based static analysis tool represented in the stockroom as an artifact, PointerBufferOverflow is a concept in the domain, also represented as a stockroom artifact, and detects is a relation in the stockroom ontology. This triple then captures the domain knowledge that one of the kinds of errors that CodeHawk can detect is a buffer overflow resulting from dereferencing a pointer.

The approach to formulating and executing queries on stockroom ontology data makes use of an ontology captured in OWL. The OWL ontology is assembled from two sources, the metadata facts about artifacts in the stockroom, represented by the taxonomic terms and triples associated with stockroom artifacts, and an OWL ontology that captures the semantics of the triple predicates. This file is combined with the facts to yield the complete ontology, which is stored in the file s3.owl. The complete ontology is then the subject of ontological queries.

The stockroom prototype supports ontological queries expressed in description logic through the use of the tool Protégé. Description logic is a logic used for describing domain concepts that is more powerful than propositional logic but not as expressive as predicate calculus, and is widely used in the ontology community because it has efficient decision procedures. The version of Protégé used with the stockroom prototype includes a description logic reasoner that can evaluate description logic queries. Referring back to the representative ontological query example in Figure 54, a similar description logic query of the stockroom can be expressed as *Static_Analysis_Tool and analyzes value C and detects value PointerBufferOverflow*. The stockroom ontology s3.owl provides sufficient information for the reasoner to produce as result of this query the *CodeHawk* stockroom artifact.

```
analyzes( CodeHawk , C )
compiles( OCP Ant , Java )
detects( CodeHawk , NullPointerError )
detects( CodeHawk , PointerBufferOverflow )
documents( OCP Documentation , OCP )
greaterThan( Flight Critical , Mission Critical )
greaterThan( Hard Real-Time , Soft Real Time )
greaterThan( Mission Critical , Not Safety Critical )
greaterThan( Safety Critical Rigor Level 2 , Not Safety Critical )
greaterThan( Safety Critical Rigor Level 3 , Safety Critical Rigor Level 2 )
greaterThan( Soft Real Time , Not Real-Time )
hasQaulityAttributeValue( OCP , Hard Real-Time )
hasQaulityAttributeValue( OCP , Mission Critical )
hasQaulityAttributeValue( SOSCOE Micro Edition , Safety Critical Rigor Level 3 )
hasQaulityAttributeValue( SOSCOE Real-Time Edition , Hard Real-Time )
hasQaulityAttributeValue( SOSCOE Real-Time Edition , Safety Critical Rigor Level 2 )
hasQaulityAttributeValue( SOSCOE Standard Edition , Soft Real Time )
leadsTo( Defect , Fault )
leadsTo( Fault , Failure )
lessThan( Mission Critical , Flight Critical )
lessThan( Not Real-Time , Soft Real Time )
lessThan( Not Safety Critical , Mission Critical )
lessThan( Not Safety Critical , Safety Critical Rigor Level 2 )
lessThan( Safety Critical Rigor Level 2 , Safety Critical Rigor Level 3 )
lessThan( Soft Real Time , Hard Real-Time )
POC( SOSCOE , Larry Culver )
POC( SOSCOE , Paul Schoen )
provides( OCP Common Services , CORBA Event Service )
provides( SOSCOE , CORBA Event Service )
requires( OCP Build Environment , GNU Make 3.79 )
requires( OCP Build Environment , GNU Make 3.8 )
requires( OCP Capstone Demonstration , OCP Framework )
requires( OCP Examples , OCP Framework )
requires( OCP Framework , OCP Build Environment )
```

Figure 55: Subset of Stockroom Triples

Figure 56 displays the representation in stockroom.owl of the predicate requires. The OWL fragment shows that requires is a transitive property between artifacts, and is a refinement of the relation depends.

```
<!-- http://www.semanticweb.org/ontologies/2009/5/Ontology1245875834530.owl#requires -->

  <owl:ObjectProperty rdf:about="#requires">
     <rdf:type rdf:resource="&owl;TransitiveProperty"/>
     <rdfs:domain rdf:resource="#Artifact_Type"/>
     <rdfs:range rdf:resource="#Artifact_Type"/>
     <rdfs:subPropertyOf rdf:resource="#depends"/>
  </owl:ObjectProperty>
```

Figure 56: Requires Predicate in Stockroom.owl

Figure 57 shows the entry in the s3.owl ontology for a representative artifact, OCP Build Files. Note s3.owl includes information about all of the artifacts associated taxonomic terms, as well as triples involving property version and relations supports and requires.

```
<!-- http://www.semanticweb.org/ontologies/2009/5/Ontology1245875834530.owl#OCP_Build_Files -->

  <Ontology1245875834530:non-EAR rdf:about="&Ontology1245875834530;OCP_Build_Files">
     <rdf:type rdf:resource="&Ontology1245875834530;Build_File"/>
     <rdf:type rdf:resource="&Ontology1245875834530;License_Not_Specified"/>
     <rdf:type rdf:resource="&Ontology1245875834530;Unclassified"/>
     <rdf:type rdf:resource="&Ontology1245875834530;non-ITAR"/>
     <Ontology1245875834530:version rdf:datatype="&xsd;float">3.0</Ontology1245875834530:version>
     <Ontology1245875834530:supports rdf:resource="&Ontology1245875834530;Cplusplus"/>
     <Ontology1245875834530:requires rdf:resource="&Ontology1245875834530;GNU_Make_379"/>
     <Ontology1245875834530:requires rdf:resource="&Ontology1245875834530;GNU_Make_38"/>
  </Ontology1245875834530:non-EAR>
```

Figure 57: OCP Build Files Artifact in S3.owl

The current stockroom ontology (as well as the taxonomies), as reflected in s3.owl, is the product of a number of factors. We examined a number of existing taxonomies and ontologies relevant to the domain, including the Association of Computing Machinery (ACM) and Institute of Electrical and Electronics Engineers (IEEE) computer science taxonomies, the Software Development Tools And Technology Software Development Tools And Technology Information Clearinghouse (SDTATIC) software tool taxonomy, DoD UAV taxonomies, as well as other relevant taxonomies and ontologies. The relevant attributes and characteristics required to describe and distinguish the set of stockroom prototype artifacts was a key driver for the final taxonomy. Stockroom operational requirements factored in other aspects of the taxonomies, in particular the licensing and export status related taxonomies. The need to support the evolution and development of stockroom capabilities was another driver in the development of the final prototype ontology.

4.2.3.3. Evolution

One of the principles of the design and architecture of the stockroom was that the ontology and taxonomies would not be static. All would evolve over the course of the lifetime of the stockroom. This was borne out even over the limited experience gained during the S3 Phase 1 prototyping effort. As indicated, there are a number of reasons that the stockroom ontology and taxonomies will change over the lifetime of the stockroom. First, the underlying domain of software for unmanned aerial systems is not static. Second, the contents of the stockroom is not static, and as more artifacts are added, in order for the taxonomies and ontology to distinguish artifacts that are similar, the taxonomies and ontology may need to be refined to capture ever finer distinctions. Finally, as the stockroom community grows and evolves, the taxonomies and ontology that the community uses will also grow and evolve. Accordingly, it is critical that the stockroom taxonomies and ontology are designed to grow and evolve.

The stockroom uses a two pronged approach to taxonomy evolution. Actual migration of the ontology and (most) of the taxonomies is performed by the stockroom administrators at the direction of a steering committee. In Phase 1, the entire Boeing stockroom team formed the steering committee. In Phase 2, the steering committee would be formed as a CSWG with limited membership. This ensures that the ontology and taxonomies remain coherent and evolve in a controlled and disciplined manner. However, the steering committee does not (and will not) have a monopoly on the domain knowledge that should drive ontology evolution. Community members can suggest changes to the taxonomies and ontology to the committee. The keywords taxonomy is a key part of the evolution strategy for the stockroom, allows and encourages input from the entire stockroom community, and is the bridge between the taxonomies and the ontology.

As discussed in Section 4.2.3.1, any user can add a new keyword by applying it to a stockroom artifact. This allows any member of the stockroom community to easily evolve the stockroom metadata. Because of the limited semantics associated with keywords, this has only a limited impact. As part of the activities of the steering committee, the current keywords taxonomy will periodically be reviewed to determine if any keywords should be “promoted” to the more structured segments of the stockroom metadata, principally the artifact type taxonomy and the ontology. In this way, ongoing stockroom experience can be input to the ontology and taxonomy evolution process, which is an emerging best practice in this field.

One of the requirements on the evolution process is that existing metadata (facts in the stockroom) must be preserved. Accordingly, one of the responsibilities of the steering committee is to define mappings from the current ontology to the new ontology in such a way that existing metadata can be readily mapped, ideally without requiring intervention by the original provider of that metadata. This ideal will not always be completely achievable. If the ontology is extended to include a completely fresh concept, then there may be no (readily) available information that would support classification of existing artifacts, in which case the owner of the artifact may be notified of an opportunity to supply additional metadata related to the fresh concept. Likewise, if an existing concept is partitioned, it may not be clear to which partition an existing artifact belongs. Tool support is required to enable the manageable migration of metadata as the ontology and taxonomies evolve.

4.2.3.4. Tools

Tool support in Phase 1 and the prototype falls into three main categories: tools for developing taxonomies and ontologies, tools for capturing and exploiting taxonomies and ontologies in the stockroom, and tools to support evolution of the stockroom taxonomies and ontology. The first category includes two freeware tools, CmapTools [10] and Protégé. The second category includes Drupal itself as well as Protégé. The third category was filled in the prototype by proof of concept tools developed during Phase 1.

CmapTools is a free tool from the Institute for Human and Machine Cognition (IHMC) for graphically modeling concept maps. Concept maps are graphs for representing concepts and the relationships between them, which is a natural way to view both taxonomies and ontologies. Figure 58 shows an early version of the Artifact Type taxonomy in CmapTools. We represented taxonomy terms as concepts (ovals). All relationships (edges) labeled ???? represent the hierarchy of the taxonomy. Other relationships (edges with other labels) were aids to the design of the taxonomy and were not necessarily reflected in the final taxonomy. For example, the *has* relationship between *Tools* and *Host Platform* evolved into the *supports* predicate in the ontology. We also used CmapTools to explore the broader stockroom ontology.

Figure 59 shows an early version of the stockroom domain ontology created in CmapTools. Ultimately, this information was reflected in the artifact type taxonomy, the stockroom predicates, and in the triples associated with the artifacts created in the stockroom. For example, the CriticalityDefinition subgraph in the lower right center of

Figure 59 eventually was realized in the various stockroom artifacts of type *Quality Attribute* reflecting different criticality levels.

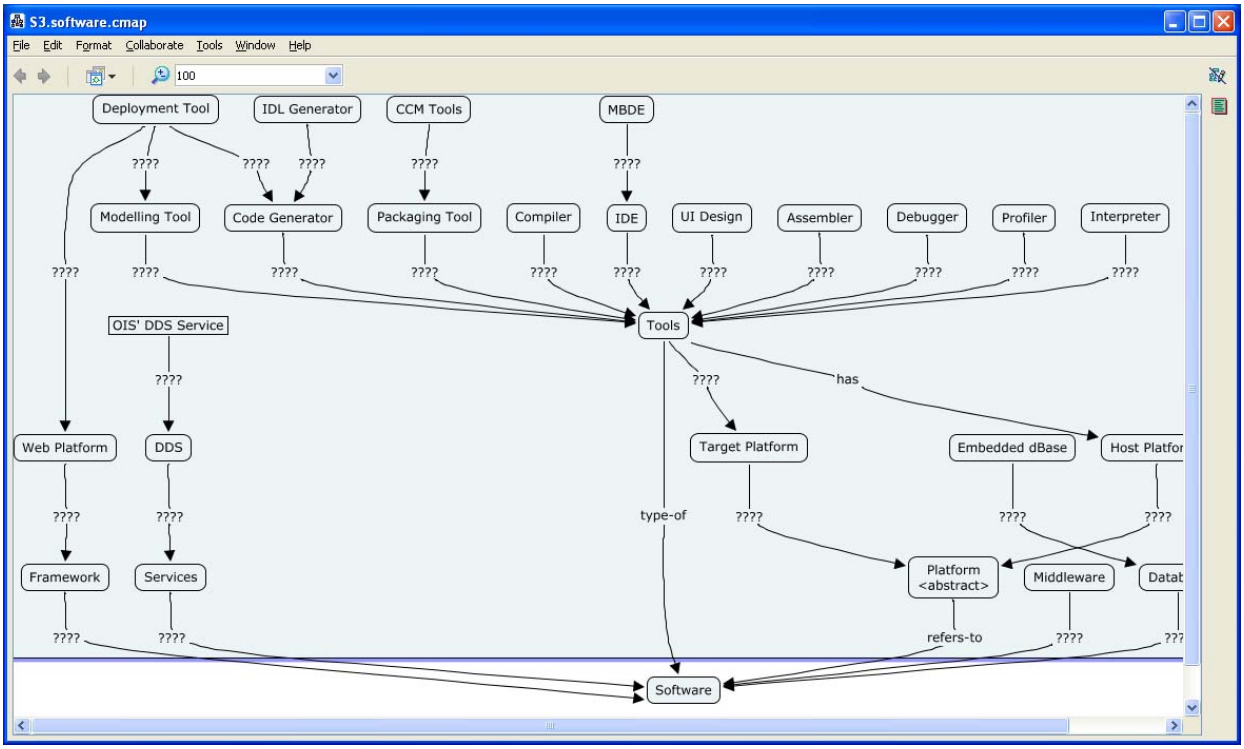


Figure 58: Early Cmap of Artifact Type Taxonomy in CmapTools

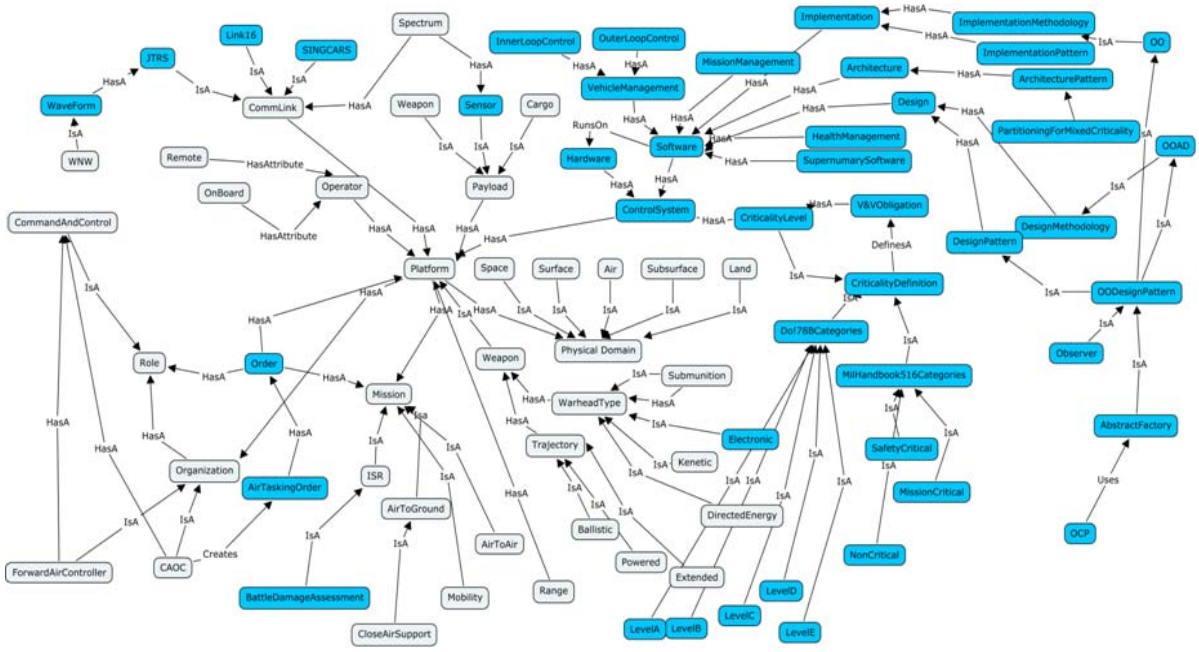


Figure 59: Early CmapTools Version of Stockroom Domain Ontology

Protégé, or more specifically Protégé-OWL 4, the specific version used, is a free open source ontology editor and knowledge-base framework developed by collaboration between Stanford University and the University of Manchester. Protégé was used both for developing the ontology, and for exploiting the ontology. The primary role of Protégé in developing the ontology was in the development of the stockroom.owl ontology that primarily servers to capture the semantics of the various stockroom predicates, as seen in Figure 56. Figure 60 shows the same information about the *requires* predicate in Protégé.

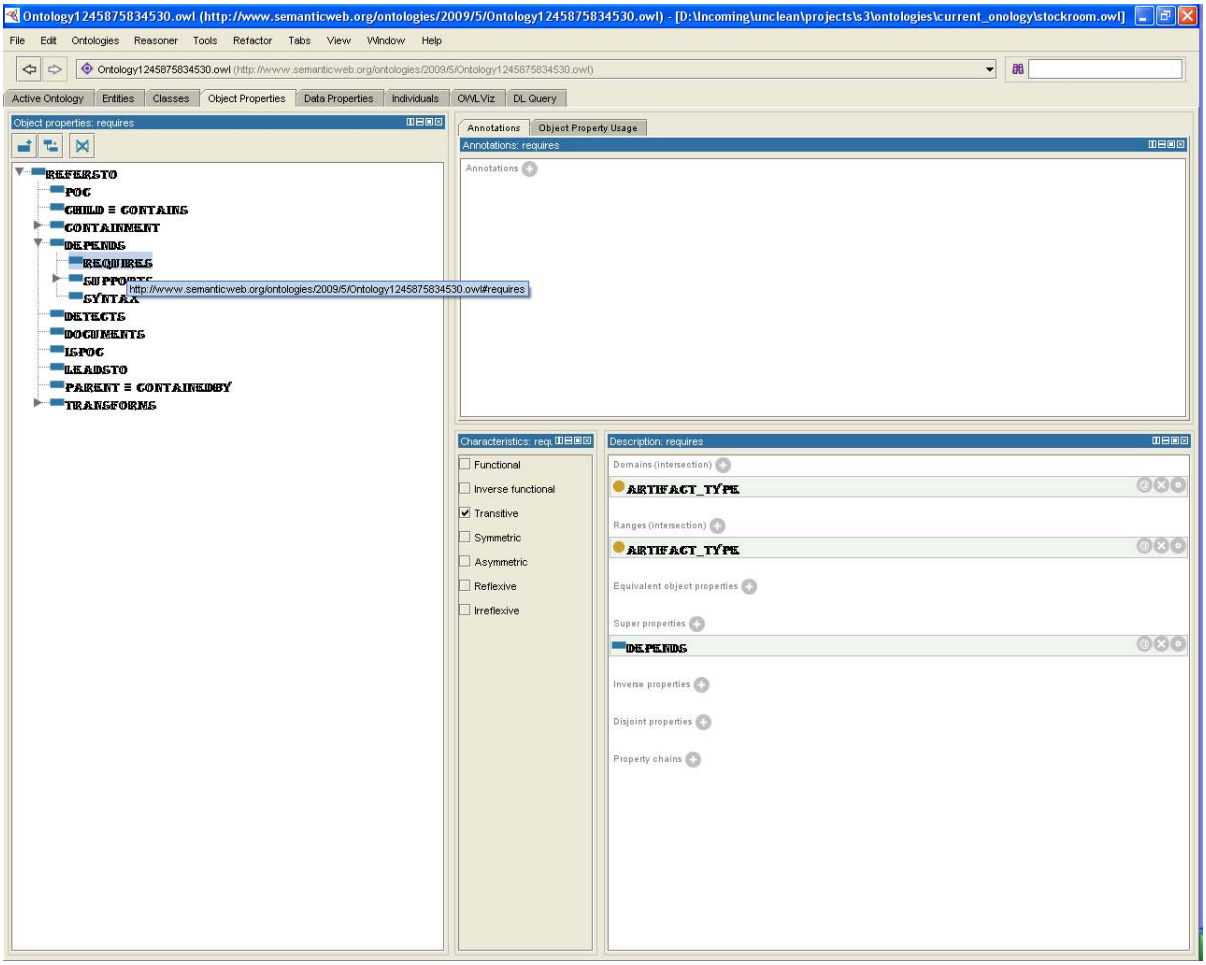


Figure 60: Stockroom.owl Ontology in Protégé

The taxonomies were represented in Drupal using Drupal's native taxonomy mechanisms. These are shown in Figure 51 and Figure 53. The ontology is represented as both artifacts, which are web pages and managed by Drupal as web content, and triples, which are stored in the Drupal Content Management System database. As indicated in Section 4.2.3.2, Protégé is used to perform ontological queries in the Phase 1 stockroom prototype. This involves the use of both Protégé and custom generated scripts. Protégé is used to actually execute the query. Figure 61 shows Protégé performing the query of Section 4.2.3.2. The scripts are used to combine the stockroom.owl ontology with the facts stored in the stockroom to produce the complete s3.owl ontology.

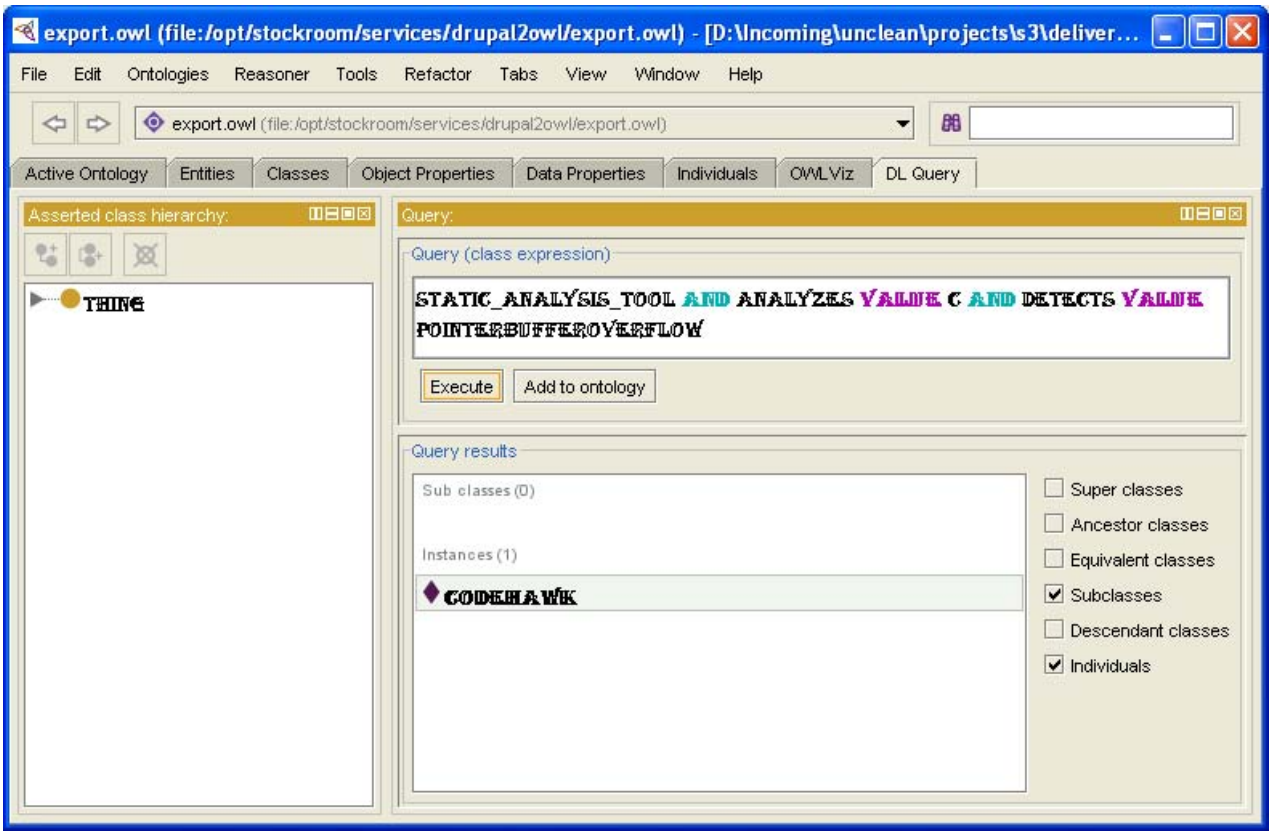


Figure 61: Description Logic Query Executed in Protégé

The final set of tools, those supporting evolution, are represented in the Phase 1 prototype by a collection of proof of concept tools developed during the program to explore the type of tool support, and approaches to providing that support, required by the evolution of the ontology and taxonomies. One such tool is a web based taxonomy manager created to support modifications to the taxonomies. The current tool allows taxonomy terms to be added and deleted, terms to be moved, and terms to be merged. Changes propagate to the artifacts associated with the terms. Figure 62 is a screenshot of the current version of the Taxonomy Manager being used to modify the Keywords taxonomy.

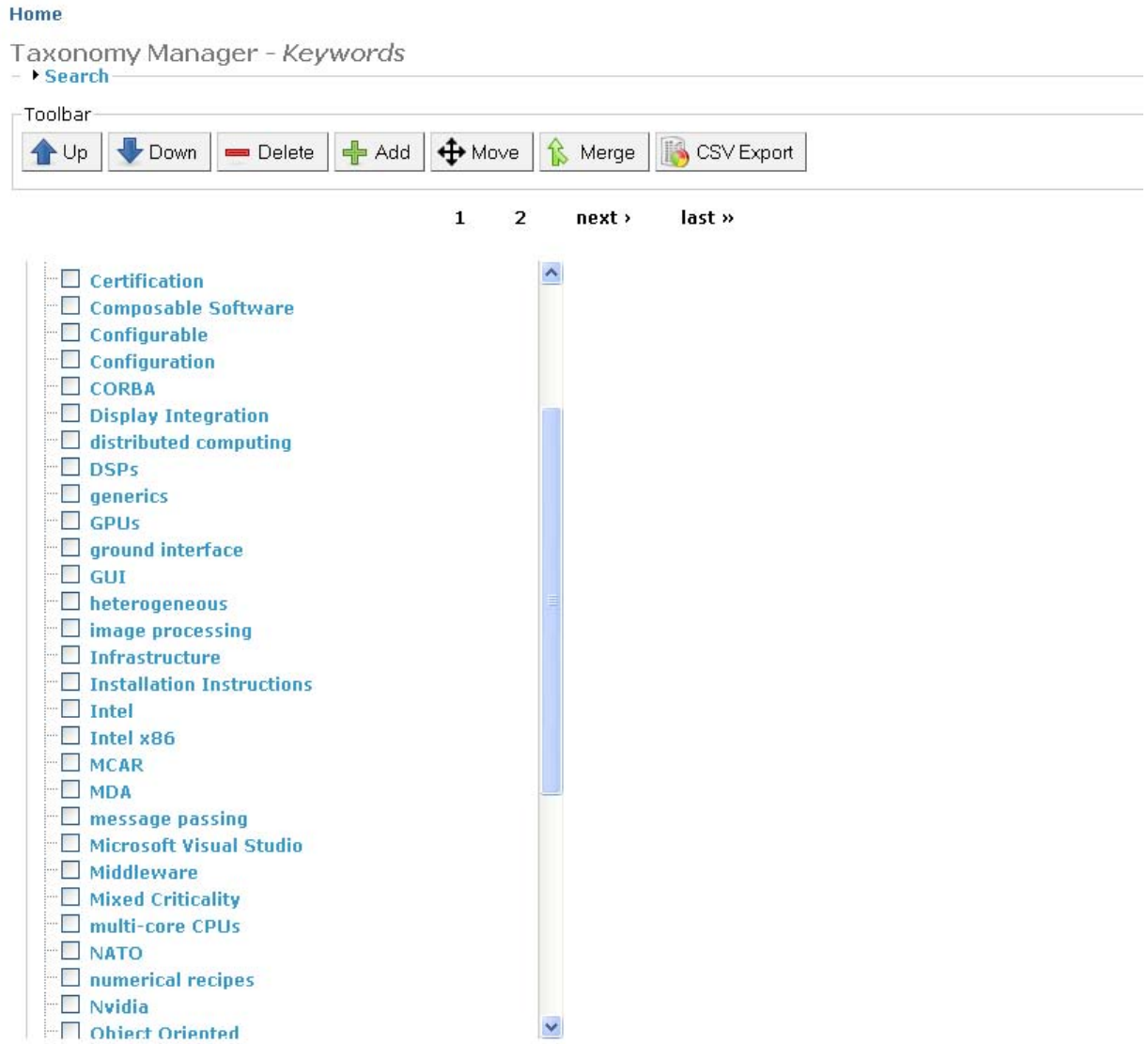


Figure 62: Taxonomy Manager Taxonomy Evolution Tool

4.2.4. Stockroom Contents

The stockroom contains two primary types of user visible content: artifacts and collaboration content. Although the ontology and taxonomies represent reusable domain knowledge and could be considered content, they have been addressed in Section 4.2.3 and will not be further addressed here. One of the primary distinctions between artifact content and collaboration content is that only artifact content is searched by metadata and ontology search. All content is searched by natural language search.

4.2.4.1. Artifact Content

Artifacts are the primary form of stockroom content and are the vehicle for the capture and re-use of domain knowledge. Artifacts can correspond to either pieces of software, documents, or other concrete entities that a stockroom user can download and use, or can be other forms of domain knowledge that are generally used to describe or provide context for other artifacts. Examples of artifacts corresponding to concrete entities include the Open Control Platform (OCP) and the Autonomous Flight Systems Stockroom User Guide. Examples of less concrete artifacts include abstract concepts such as UAV, Fault, and Criticality Level, or concrete entities that are not themselves available through the stockroom, such as C++ (though a particular standard document or compiler might be available through the stockroom), the Global Information Grid, and the RMax unmanned rotorcraft. The stockroom prototype contains approximately 125 artifacts.

Three of the major artifacts in S3 represent reusable software platforms that would be useful in the autonomous flight systems embedded software domain – the Signal Processing Platform (SPP), the OCP, and the System-of-Systems Common Operating Environment (SOSCOE). For each of these platforms, the data available through S3 includes:

- framework software
- software developer tools
- developer documentation
- example programs to help the software developer learn how to use the platform

The remainder of this section provides additional information about these platforms and other concrete artifacts.

4.2.4.1.1. SPP

The SPP is a tool-chain, schematically represented in Figure 63, which evolved and is maintained by Vanderbilt University through a series of Raytheon funded university research grants. It leverages work previously developed within the Defense Advanced Research Project Agency (DARPA) Adaptive Computing Systems (ACS) and Model-Based Integration of Embedded Software (MoBIES) programs. SPP implements a model-based design process to design signal processing applications running on heterogeneous, embedded platforms, composed of high-performance Field Programmable Gate Arrays (FPGAs) with embedded processor cores, Application Specific Integrated Circuits (ASICs), Digital Signal Processors (DSPs), and Reduced Instruction Set Computer (RISC) processors.

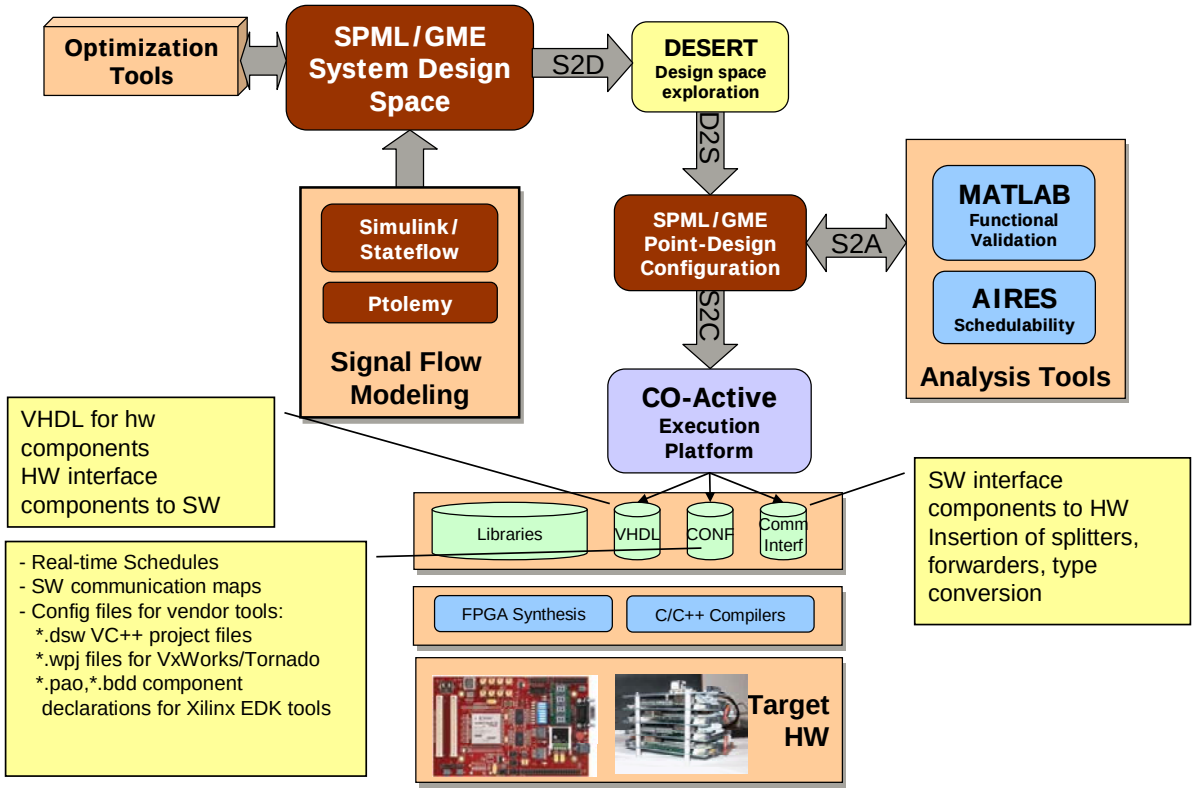


Figure 63: Signal Processing Platform (SPP) Tool-chain

The tool chain consists of:

- Component and System Modeling Tool that supports creation of models that describe the embedded platform - FPGAs, DSPs, ASICs, and General Purpose Processor s (GPPs), component allocations, etc. A key feature of the tool is the ability to model design alternatives, capturing a design space in the form of hierarchically arranged alternatives.

- Design Space Exploration Tool (DESERT) allows exploration and pruning of the large design spaces that are captured, based on user-supplied performance and resource constraints. This gives the ability to obtain a custom solution from a generic and (potentially) large design space based on the specific requirements in a largely automated manner. It also provides an automated way of partitioning functionality across the hardware-software boundary.

- Design Simulation Tool (Simulink/Stateflow, Ptolemy) supports a functional simulation of the signal processing application.

- Design Analysis Tools (MATLAB, AIRES) supports event and timing analysis (including schedulability).

- CoActive Build and Execution platform supports the synthesis of hardware and software, and the interface for communication across the hardware/software boundary.

- A suite of modeling support and utility plug-ins that facilitate the construction, manipulation, and management of large models.

Figure 64 lists platforms currently supported by SPP.

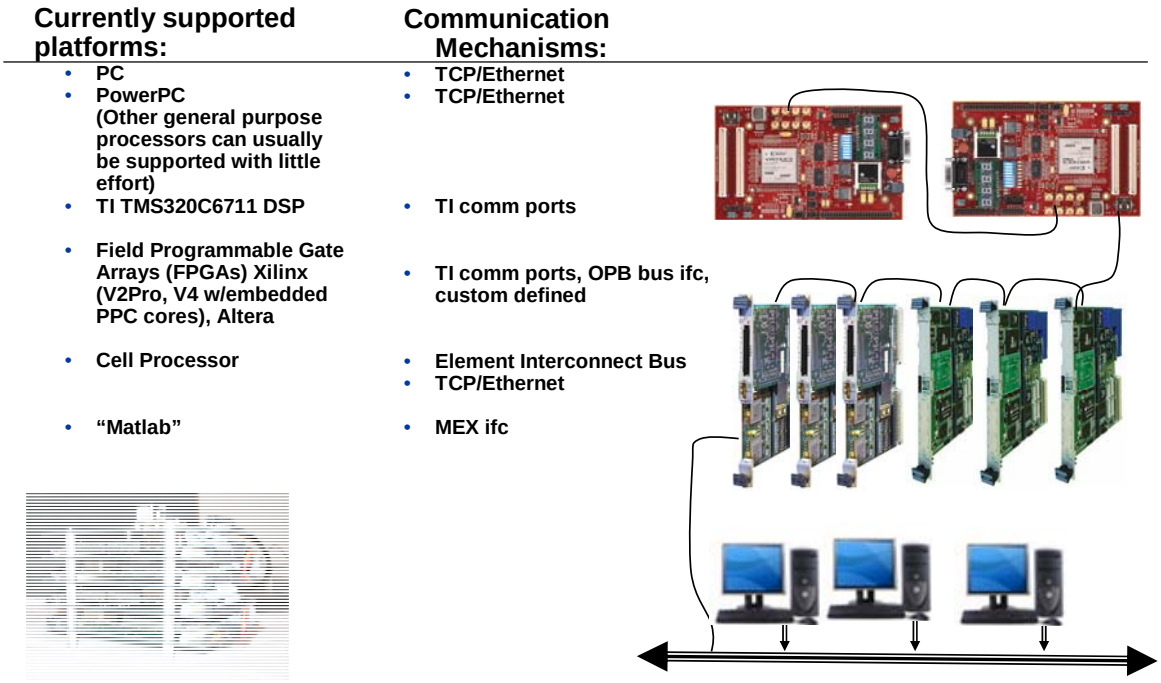


Figure 64: SPP – Supported Platforms

As previously mentioned, the SPP is maintained by Vanderbilt University and is distributed as open-source software via the ESCHER web portal. (Detailed, web-based SPP user documentation is also maintained on the ESCHER site as well.) Due to this preexisting distribution mechanism, it was decided to leverage the *Contribute with data reach-back* use case, and model the Stockroom SPP artifact with a URL link to the remote data payload (on the ESCHER web portal). Aside from this, the SPP artifact is identical to artifacts with local data payloads. The SPP is modeled on Stockroom using two artifacts – a primary artifact containing a URL to the SPP distribution web page (on ESCHER) and a child artifact for the SPP documentation, with URL to the SPP documentation home page.

The SPP tool-chain facilitates the rapid prototyping of high performance embedded computing problems, and in particular may address programming challenges related to the emerging class of multi-core processing architectures. It should be noted that the SPP distribution includes several (working) example models, including a correlation-based Automatic Target Recognition (ATR) algorithm, and software radio examples.

4.2.4.1.2.OCP

The OCP was developed on the DARPA/AFRL Software Enabled Control (SEC) program as a multi-platform software infrastructure for advanced control technologies for unmanned fixed and rotary winged aircraft. Specifically, the OCP provided a platform that allowed controls engineers to focus on control algorithms, while the OCP provided scheduling, data distribution and the other software infrastructure services required on flight platforms. The OCP was based on the Boeing Bold Stroke infrastructure that provided a similar platform for a number of Boeing production platforms, including the F-15, F-18, and AV-8B, as well as the Boeing X-45A J-UCAS. The OCP itself has flown on F-15, the Boeing T-33 J-UCAS surrogate, and Boeing/Insitu ScanEagle fixed wing aircraft, and RMax and Maverick A-160 surrogate rotorcraft.

The OCP includes the actual OCP framework software that would be deployed to an embedded platform, tools to architect and generate a system that uses the OCP, tools for building the OCP, and documentation. The OCP architecture includes a component based application layer, a component infrastructure, a collection of services tailored to the avionics domain, and COTS middleware and abstraction layers. Figure 65 shows the layers of the OCP infrastructure, and Figure 66 shows the component structure of a sample OCP application.

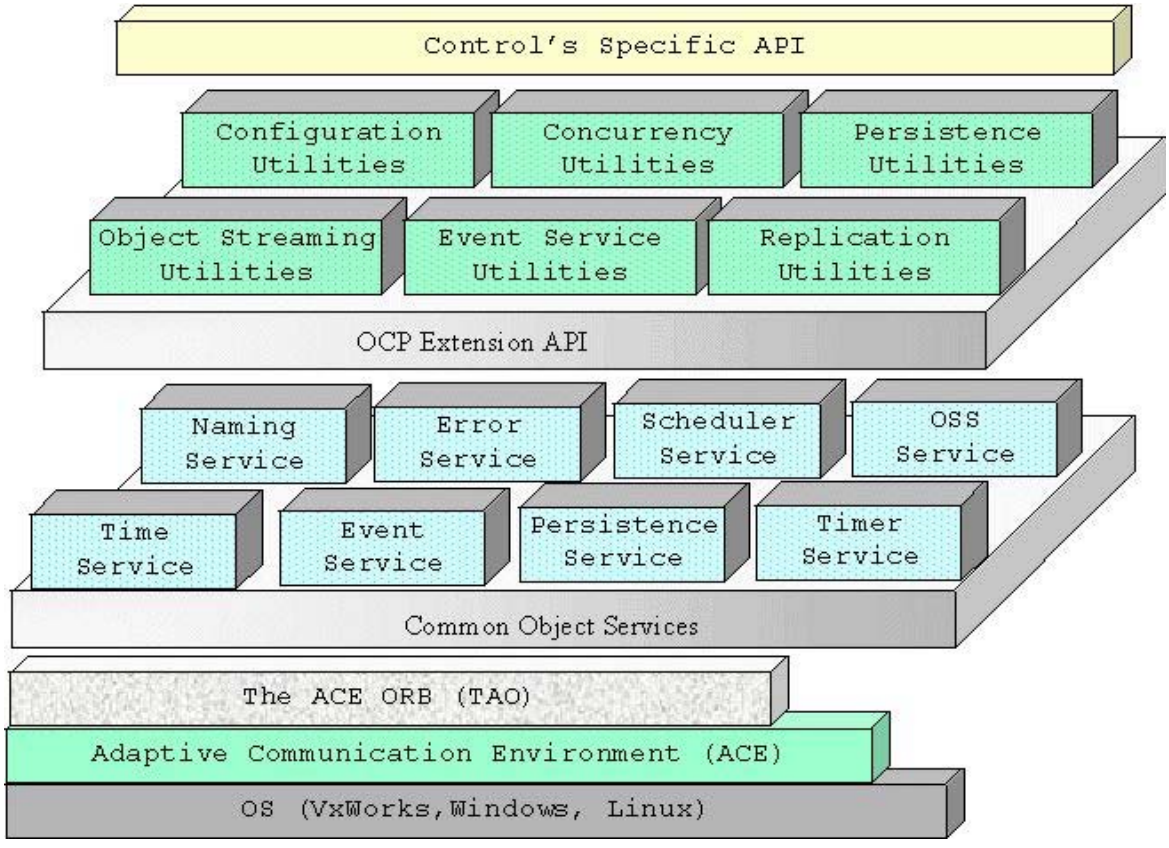


Figure 65: OCP Infrastructure

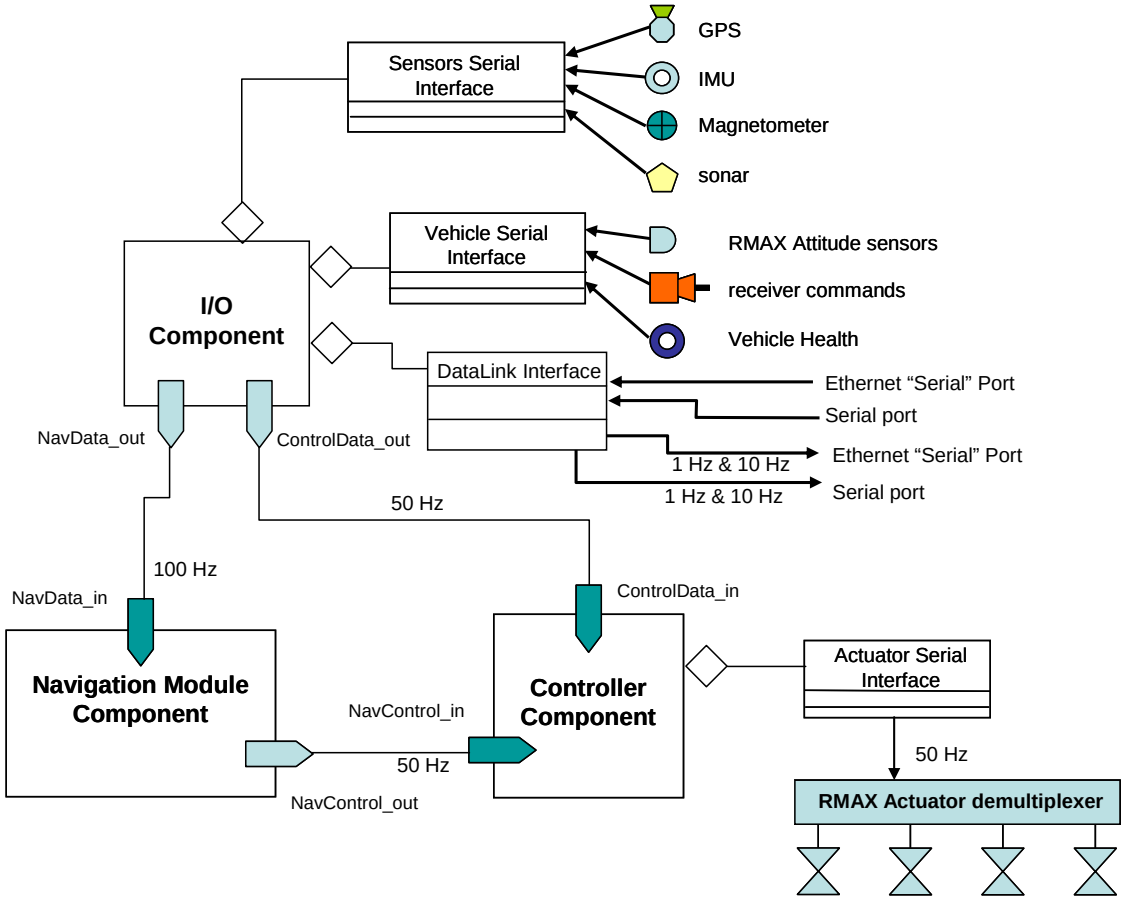


Figure 66: Sample OCP Application

The OCP is represented in the stockroom by over twenty different artifacts of various types. Infrastructure components include *OCP Common Services*, which includes the common service abstractions, and *OCP Platform Services*, which includes the COTS middleware and abstraction layers. The *Controls API Tool* is the OCP tool for generating the component skeletons and configuration and build information for an application using the OCP. *OCP Development Environment Tools* and *OCP Ant* are among the artifacts that form the OCP build environment, and the *OCP Documentation* artifact contains the full set of OCP documentation, including user's guide and installation instructions. The *OCP Examples* and *OCP Capstone Demonstration* artifacts contain example component based applications built using the OCP. These are all concrete artifacts that have attached payloads. *OCP Framework* is a more abstract artifact that represents the OCP application framework, and serves as the parent artifact for the various concrete artifacts that make up the runtime OCP infrastructure.

4.2.4.1.3.SOSCOE

The S3 team coordinated with the Boeing Combat Systems organization to allow the Army Future Combat Systems (FCS) SOSCOE platform (see Figure 67) to be made available on S3. SOSCOE is available free of charge for DoD programs, and access to the software, which is ITAR-restricted, must be coordinated with appropriate Boeing and US Army personnel. Due to the ITAR restrictions on SOSCOE, the payload is not resident on S3, but rich metadata describing the platform and its utility is hosted on S3 along with appropriate point-of-contact information for download.

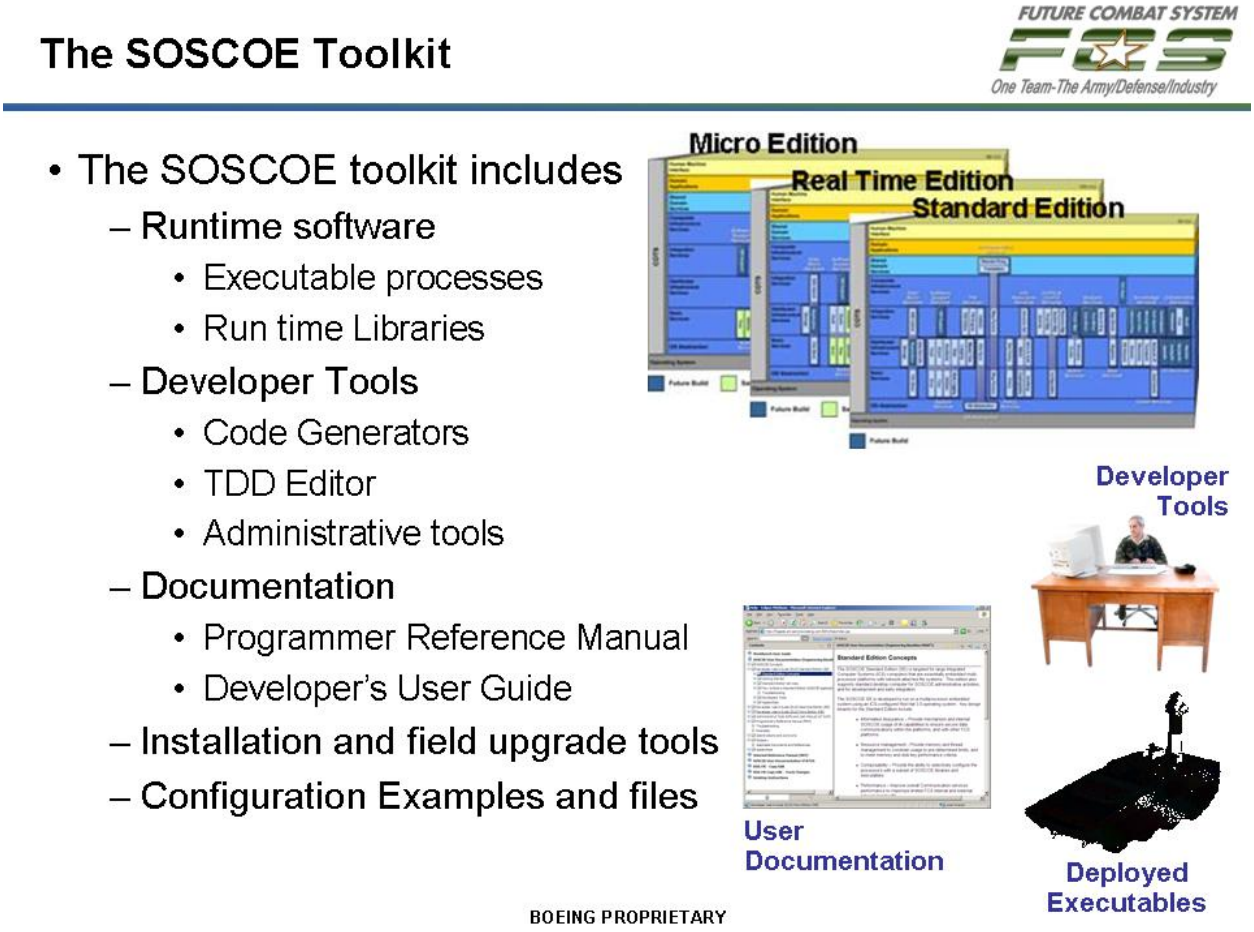


Figure 67: SOSCOE Artifact Contents

There are three “Editions” of SOSCOE that are available through S3:

(1) Standard Edition – contains the most services of all SOSCOE Editions; cannot run hard real-time; not for resource-constrained systems (appropriate for UAV control stations); known to work on a single CPU Intel x86 Pentium processor running RedHat Enterprise Edition.

(2) Real-Time Edition – Has fewer services than Standard Edition and has specialized capabilities for supporting real-time embedded systems. Executes with a real-time operating system (RTOS); key design considerations include performance, resource constraints, footprint, and composability; known to work on Radstone’s CP1A single board computer; applicable to on-board UAV processing in mid- to large-sized UAVs.

(3) Micro Edition – enough functionality to enable network interoperability for highly embedded applications with small processors, low memory, and power consumption restrictions; known to work on TI digital signal processor 6713; applicable to mini and micro UAVs, and small Unattended Ground Sensors.

4.2.4.1.4.Embedded Processing Technology Content

To further demonstrate the ability of the stockroom to represent artifact metadata, including taxonomic classification and ontological relationships, a number of embedded processing technology artifacts were pre-populated into the stockroom by the Boeing team. In particular, a variety of multi-core artifacts representing software technologies from the academic and COTS vendor communities were created. URLs with reach-back to providers’ web sites were included, as were RDF triples capturing dependency and supportability information for the respective technologies.

It is envisioned that some members from this community will become members of the stockroom and will eventually wish to actively manage the artifact metadata of their respective technologies. However, even in the absence of such participation, it is possible for interested parties (e.g., S3 Administration, TWG and/or CSWG representatives) to capture and maintain artifact metadata of this type. It is expected this type of knowledge capture and representation will be commonly practiced once the stockroom is fully deployed in Phase 2.

4.2.4.1.4.1. Embedded Processing Technology Content Descriptions

The following are brief descriptions of the embedded processing technology artifacts pre-populated into the stockroom:

CUDA - (originally an acronym for Compute Unified Device Architecture, although this is no longer used) is a parallel computing architecture developed by NVIDIA. Simply put, CUDA is the computing engine in NVIDIA graphics processing units (GPUs) that is accessible to software developers through industry standard programming languages. Programmers use C for CUDA (C with NVIDIA extensions), compiled through a PathScale Open64 C compiler, to code algorithms for execution on the GPU. CUDA architecture supports a range of computational interfaces including OpenCL and DirectX Compute. Third party wrappers are also available for Python, FORTRAN and Java.

RapidMind - a development and runtime [software] platform that enables single threaded manageable applications to fully access multi-core processors. With RapidMind, developers continue to write code in standard C++ and use their existing skills, tools and processes. The RapidMind platform then parallelizes the application across multiple cores and manages its execution.

Gedae - a model-based development environment which supports automatic multi-threaded implementations for multi-core and multi-processor systems. It includes Integrated Development Environment (IDE), programming language with graphical and symbolic UIs, compiler, deployment support tools, static and dynamic analysis tools.

StreamIt - a programming language and a compilation infrastructure, specifically engineered for modern streaming systems. It is designed to facilitate the programming of large streaming applications, as well as their efficient and effective mapping to a wide variety of target architectures, including commercial-off-the-shelf uni-processors, multi-core architectures, and clusters of workstations.

Mercury MultiCore Framework – an Application Programming Interface (API) for applications based on the Cell Broadband Engine™ (BE) processor. It maximizes resources and application performance by taking full advantage of the multi-core processor's computation model. The MCF library of functions manages concurrent processes and efficiently distributes data.

Mercury MultiCore Scientific Algorithm Library - enables applications to achieve higher performance from advanced multi-core processors. Source-code compatibility with existing applications helps decrease time-to-market by minimizing the changes to existing applications as embedded systems adopt multi-core processors.

IBM SDK for [Cell] Multi-core Acceleration IDE - built upon the Eclipse and C Development Tools platform. It integrates the Cell/B.E. GNU tool chain, compilers, IBM Full-System Simulator (simulator) for the Cell/B.E. processor, and other development components to provide a comprehensive development platform that simplifies Cell/B.E. application development.

Data Communication and Synchronization library - provides a set of services which ease the development of applications and application frameworks in a heterogeneous multi-tiered system, for example a 64 bit x86 system (x86_64) and one or more Cell/B.E. systems. The services are implemented as a set of APIs providing an architecturally neutral layer for application developers on a variety of multi-core systems. It is included with the IBM Cell SDK (Software Development Kit).

Accelerated Library Framework - provides a programming environment for data and task parallel libraries and applications. (It is included with the IBM Cell SDK.)

Charm++ - a machine independent parallel programming system. Programs written using this system will run unchanged on MIMD machines with or without a shared memory. It provides high-level mechanisms and strategies to facilitate the task of developing highly complex parallel applications. Programs are written in C++ with library calls and an interface description language for publishing Charm++ objects. Charm++ supports multiple inheritance, late bindings, and polymorphism.

Cilk++ is three keywords and a runtime system that extend C++ to the realm of multicore (or parallel) programming. The platform-independent Cilk++ solution addresses the key multicore programming challenges of development time, software reliability, and performance.

Intel Integrated Performance Primitives - extensive library of multi-core-ready, optimized software functions for digital media and data-processing applications. It offers thousands of optimized functions covering frequently-used fundamental algorithms. Its functions are designed to deliver performance beyond what optimized compilers alone can deliver.

Intel Math Kernel Library - library of optimized, extensively threaded math routines for science, engineering, and financial applications that require maximum performance. Core math functions include Sparse Solvers, Fast Fourier Transforms, Vector Math, and more. Offering performance optimizations for current and next-generation Intel® processors, it includes improved integration with Microsoft Visual Studio, Eclipse, and XCode. It allows for full integration of the Intel Compatibility OpenMP run-time library for greater Windows/Linux cross-platform compatibility.

Intel Threading Building Blocks - offers a rich and complete approach to expressing parallelism in a C++ program. It is a library that helps you take advantage of multi-core processor performance without having to be a threading expert. Threading Building Blocks is not just a threads-replacement library. It represents a higher-level, task-based parallelism that abstracts platform details and threading mechanism for performance and scalability and performance.

VSIPL++ Library - [specification] provides C++ classes and functions for writing embedded signal processing applications designed to run on one or more processors. VSIPL++ contains

- containers such as vectors, matrixes, and tensors,
- mathematical operations such as addition and matrix multiplication on these containers,
- complex numbers and random numbers,
- linear algebra solvers such as LU Decomposition, and
- signal processing classes and functions including fast Fourier transforms, convolutions, correlations, Finite Impulse Response (FIR) filters, and Internet Routing Registry (IIR) filters.

VSIPL++ extends the Vector, Signal, and Image Processing Library (VSIPL) standard, improving application performance, productivity, and portability. This C++ library supports improved performance through support for execution using multiple processors and through improved code optimization. The ability to write expressions using mathematical notation decreases program size, improving productivity.

PVTOL (Parallel Vector Tile Optimizing Library) - portable and scalable middleware library for multi-core processors, which enables incremental development. PVTOL project has two goals: 1) Create a library that allows multi-core processors to be quickly programmed for real signal processing applications; 2) Develop prototype multi-core VSIPL extensions for possible adoption by the VSIPL standards body. The main features of PVTOL are:

- Parallel VSIPL++ look and feel: scalable, portable, high performance.
- Hierarchical Arrays.
- Automated Parallel Mapping.
- Heterogeneous Processors Dispatch Mechanism

OpenMP (Open Multi-Processing) - an API that supports multi-platform shared memory multiprocessing programming in C, C++ and FORTRAN on many architectures, including UNIX and Microsoft Windows platforms. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior.

OpenCL (Open Computing Language) - framework for writing programs that execute across heterogeneous platforms consisting of Central Processing Units (CPUs), GPUs, and other processors. OpenCL includes a language (based on C99) for writing kernels (functions that execute on OpenCL devices), plus APIs that are used to define and then control the heterogeneous platform. OpenCL provides parallel programming using both task-based and data-based parallelism.

MPI (Message Passing Interface) - Written by the MPI Forum (a large committee comprising of a cross-section between industry and research representatives), MPI is a standardized API typically used for parallel and/or distributed computing. The MPI standard is comprised of 2 documents: MPI-1 (published in 1994) and MPI-2 (published in 1996). MPI-2 is, for the most part, additions and extensions to the original MPI-1 specification.

AMPI (Adaptive MPI) - an implementation of the MPI library, that supports dynamic load balancing and multithreading for MPI applications. AMPI is built upon Charm++, a parallel C++ library.

Sieve C++ Multicore Programming System - scalable programming system aimed at those who need to create C++ code suitable for use on a multi-core processing environment. The Sieve System consists of an extension to a C++ compiler, a multi-core linker and a runtime to schedule the processes.

4.2.4.2. Collaborative Content

Multiple TWGs were stood up in Phase 1 shortly after S3 was made available as a password-protected entity on the public Internet. This has allowed us to quickly support collaboration Use Case capabilities that were used in Phase 1 to:

- get users on S3,
- get feedback from users on the web interface and collaboration capabilities, and
- provide needed collaboration capability for the mil-aero industry to support current work spanning multiple R&D programs.

More detailed information on these Phase 1 TWGs is provided below.

4.2.4.2.1. Public TWGs

The Phase 1 stockroom prototype includes non-artifact WG content from four TWGs: the High-Confidence Design TWG, the Formal Methods TWG, the Mixed Criticality Architecture Requirements (MCAR) TWG, and the Transportation TWG. These TWGs were used during Phase 1 to experiment with different approaches to TWGs and stockroom enabled collaboration, and to provide additional opportunities for community engagement and awareness. The High Confidence Design TWG was stood up in anticipation of being used to support the HCSS program committee, but was never actively used.

The Formal Methods TWG was established to support the development of a presentation at the National Security Agency sponsored High Confidence Software and Systems (HCSS) Workshop. The presentation reviewed experiments conducted with two formal methods based Verification and Validation (V&V) tools during the Certification Techniques for Advanced Flight Critical Systems Challenge Problem Integration program, an AFRL/RBCC sponsored effort to demonstrate the effectiveness of advanced V&V technologies in the improvement of the V&V and certification processes as applied to software subject to airworthiness certification.

The MCAR (Mixed Criticality Architecture Requirements) TWG was established to support the MCAR program participants in development of a white paper describing the domain of mixed criticality systems and research issues associated with this architecture and for use by the program committee planning the MCAR workshop at the 2009 IEEE Real-Time and Embedded Applications Symposium. The MCAR program is an AFRL/RBCC sponsored program defining the high confidence real-time operating system and middleware requirements for next generation UAVs that have an architecture combining elements of different safety criticality levels on the same compute platform. The MCAR team, including representatives from AFRL, Boeing, Lockheed Martin, Northrop Grumman, Honeywell, LynuxWorks and Washington University in St. Louis, used the TWG to create and share drafts of the white paper, and to distribute submissions to the workshop among program committee members.

The Transportation TWG was established to coordinate writing of the Participants Report from the Nov 2008 High Confidence Software and Systems (HCSS) National Workshop for New Research Direction for High Confidence Transportation Cyber-Physical Systems (CPS) organized by the Networking and Information Technology Research & Development (NITRD) HCSS Coordinating Group. This report summarized the perspectives of the participants on the

workshop, including capturing research roadmaps for the Verification & Validation, Mixed Criticality, Platforms and Infrastructure, Autonomy and Control, and Infotronics and Infotainment aspects of CPS. The TWG was used to coordinate drafting of this Participants' Report that was provided to the Networking and Information Technology Research and Development (NITRD) HCSS Coordinating Group for inclusion in the final Workshop Report that would ultimately be provided to policymakers. The Transportation TWG included participants from Arizona State University, the University of Pennsylvania, Ford Motor Company, Boeing, the University of Washington, Carnegie Mellon University, the University of Arizona, General Motors, Georgia Tech and Washington University in St. Louis.

4.2.4.2.2. Admin TWG

The S3-Team TWG was used by the Boeing S3 team for internal collaboration during Phase 1. The stockroom collaboration capabilities established for the S3-Team TWG were used to support preparation for program meetings, including the Midterm and Final Reviews. The TWG was also used in the development of the stockroom taxonomies and ontology, including both defining the approaches for defining the taxonomies and ontology, selecting the tools to use, and creating the initial taxonomies and ontology. The S3-Team was also used in the development of a number of program deliverables, including this report. In Phase 2, a separate CSWG will be established for the Ontology Evolution Steering Committee (Section 4.2.2.2.1.2), and those activities will move out of the S3-Team TWG. Other S3-Team TWG activities will continue in Phase 2.

4.3. Community

Our S3 development team has recognized that community involvement is crucial to the success of S3. With this in mind, we have been active in engaging interest among academia and industry in participation in S3 activities and making them aware of the potential benefits of the S3 concept. Our involvement on other programs and efforts, with multiple classes of stakeholders in the autonomous flight systems embedded software domain, have allowed us to get visibility for S3 and get these stakeholders to actually use the S3 itself during Phase 1. These various classes of stakeholders are shown in Figure 2.

Many of these stakeholders actually became S3 users and community members in Phase 1 with leverage of the S3 TWG functions for collaborative development of research products. In this way, a diverse group of large and small contractors, academia, and government became part of the S3 community and leveraged the S3 to perform useful work.

Our identification of potential community members began before Phase 1 began, as shown in Table 2.

Table 2. Community Members Identified at Proposal Time

General Category	Participants
Government Acquisition, Development , and Services Program	Navy / Marine Corps Small Tactical Unmanned Aircraft System (STUAS) / Tier II, Army Future Combat Systems, Navy / Marine Corps / Air Force ScanEagle, DARPA Vulture, Unmanned Combat Air System (UCAS), Air Force Next Generation Unmanned Aerial System (NG UAS), Navy Tomahawk, Air Force Miniature Air-Launched Decoy (MALD)
Government Research Customers	OSD, ONR, AFRL, ARL, DAPRA, NSF
Small Business / Small Contractors	Insitu Group, Swift Engineering, Scientific Systems, Barron Associates, Defense Research Associates, NextGen Aeronautics, Draper Laboratory, Tech Team Government Solutions, EDAptive Computing, WW Technology, Kestrel Technology
Government, Industrial and University Research Institutions	NSF Centers for Embedded Systems and Autonomic Computing, HRL Laboratories, Vanderbilt University, University of California Berkeley, Georgia Tech, MIT, Arizona State, University of Arizona, University of Maryland, Cal Tech, University of Minnesota, University of Illinois, University of Colorado, Cornell University, Naval Postgraduate School
Commercial Software and Software Tools Vendors	IBM, Microsoft, MathWorks, Real-Time Innovations, Objective Interface Systems, Mentor Graphics, Prism Tech, Wind River Systems, Green Hills Software, LynuxWorks
Avionics and Embedded Computer Systems Manufactures	Honeywell, Mercury Computer, Curtiss Wright, GD-AIS
Processor Manufactures	IBM, Intel, TI, AMCC, Xilinx, Altera

Many of these organizations became S3 community members in Phase 1, and others expressed interest in participating in S3, including:

- Industry
  - Mil-Aero – BAE Systems, Boeing, Lockheed Martin, Northrop Grumman, Honeywell, Rockwell Collins
  - Software Providers – Wind River Systems, Green Hills Software, LynuxWorks, Objective Interface Systems, Real-Time Innovations
  - Tools – Math Works, Kestrel Technology, EDAptive Systems, WW Technology Group
  - Processor Manufacturers – Intel (initiated)
- Academia
  - Arizona State, Georgia Tech, Naval Postgraduate School, Univ. of Arizona, Washington University, University of Washington

Our community building activities also included pro-active participation in national conferences to get visibility for S3. Each of these briefings was done in front of an audience that included a mix of industry, academia, and government. These national conference briefings included the following:

AIAA InfoTech@Aerospace Conference, 6-9 Apr 2009, Seattle, Washington – As a member of the American Institute of Aeronautics and Astronautics **(**AIAA) Software Systems Technology Committee (SSTC), S3 PI Jim Paunicka was able to secure a spot on the agenda for an S3 briefing at this conference. S3 was briefed at a public presentation session and was discussed at length in an AIAA SSTC committee meeting. Committee members working space vehicle systems showed keen interest in the Stockroom concept, including an AFRL representative pursuing Plug-and-Play space systems and a BAE Systems engineer also working in that area. During the conference, Jim Paunicka met with a number of academics who showed interest in hosting domain knowledge on S3. An industry representative at the public session commented on the issue of keeping contents searchable as volume of contents grows. Another industry audience member followed up with an email expressing interest "in support software for space processors, multicore software, FPGA firmware and of course plug and play". A Government audience member commented on the benefits of allowing users to comment on repository contents, which could be potentially valuable feedback.

IEEE Real-Time and Embedded Technology and Applications Symposium, 13-16 Apr 2009, San Francisco – As a member of a workshop steering committee for this symposium, S3 PI Jim Paunicka was able to secure a slot for a public briefing of S3.

AFRL Safe and Secure Systems and Software Symposium, 2-4 Jun 2009, Dayton Ohio – with the assistance of our S3 AFRL customer, we were able to brief the S3 concept to this national audience also.

In summary, significant outreach to a diverse and large number of potential community members was performed in Phase 1.

5.0 CONCLUSIONS

The S3 Phase 1 prototyping work by the Boeing, Raytheon, and Vanderbilt University team has resulted in a rapid build-up of a feature-rich repository capability. Many of the functional objectives of the repository are already in place, and feasible plans for maturation in a Phase 2 have been explored. As part of the Phase 1 prototype development, the powerful but general Drupal content management system has been augmented with domain-specific taxonomies and ontologies, tailoring the solution to the DoD domain of autonomous flight systems embedded software on emerging processing systems.

The Phase 1 S3 repository has been augmented with useful repository contents, including 125 artifacts supporting the autonomous flight systems domain. In the development and operations activities of Phase 2, we plan to evolve the S3 capabilities and domain-specific ontology, and continue building the repository content collection.

Significant outreach and engagement with potential S3 community members was also accomplished in Phase 1. Numerous talks at national conferences were delivered to increase the visibility for the S3 concept and our domain. Groups of academic, industry, and government people became active users of our S3 with the TWG concept, allowing these users to become familiar with the S3 system and concepts. We engaged many of our research and business associates, and they are willing to be active participants in the S3 should it move forward. These include representatives from the mil-aero industry, as well as other associated industries (embedded software providers, tools developers, etc.), small businesses, and academia.

Our targeted domain of autonomous flight systems is an important one for the DoD. These systems represent one of the few growth areas in platform development, so a S3 investment in this domain should be strongly considered. The complexity of the on-board software on these platforms represents a growing challenge that could be addressed with the mechanisms for capture and reuse of domain knowledge and expertise that S3 can provide.

Aspects of the supported domain associated with hosting embedded flight software on emerging processing systems (e.g., multi-core) are also significant. With more on-board capabilities desired on aircraft, this element of the S3 domain can provide important support for those needing help solving multi-core development issues.

We believe we have a strong and dedicated team, ready for the challenges of supporting this domain in a Phase 2 development and operations effort. Our team that includes two developers of autonomous flight systems that are sometimes competitors, Boeing and Raytheon, provides domain breadth and supports development of an S3 that is not dominated by a single contractor. The academic component of our team with Vanderbilt University allows us to tap their knowledge of the most recent developments in web and content management technologies, and provides a contractor-neutral organization for hosting S3 (making S3 more inviting to other contractors).

With a strong team and a credible Phase 1 effort to rapidly prototype a feature-rich S3, we look forward to an opportunity to support the autonomous flight systems domain with a Phase 2 development and operations effort.

6.0 REFERENCES

[1] McKeever, William, “Software Systems Stockroom (S3)”, briefing at the Safe & Secure Systems & Software Symposium, Dayton, Ohio, 02 Jun 2009.

[2] The Drupal Content Management System, http://drupal.org/

[3] McGuinness, Deborah L. and van Harmelen, Frank, “OWL Web Ontology Language Overview,” W3C Recommendation 10 February 2004 http://www.w3.org/TR/2004/REC-owl-features-20040210/

[4] Horridge, Matthew, “A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools Edition 1.2,” The University of Manchester, 13 Mar 2009, http://protege.stanford.edu/.

[5] Dmitry Tsarkov, “FaCT++”, University of Manchester, http://owl.man.ac.uk/factplusplus/.

[6] The Boeing Company, “Users’ Guide for S3 Operation”, Software Systems Stockroom Phase 1 CLIN 0002 CDRL Data Item No. A004, 31 Jul 2009.

[7] The Boeing Company, “Administration & Installation Guide S3 Operation”, Software Systems Stockroom Phase 1 CLIN 0002 CDRL Data Item No. A003, 31 Jul 2009.

[8] VMware, Inc., “VMware Player 2.5 Getting Started Guide,” 2008, http://www.vmware.com/pdf/vmware_player250.pdf.

[9] Sun Microsystems, “MySql Reference Manual,” http://www.mysql.com/.

[10] Institute for Machine and Human Cognition, http://cmap.ihmc.us/conceptmap.html.

7.0 LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS

ACL	Access Control List
ACS	Adaptive Computing Systems
AFRL	Air Force Research Laboratory
AFRL/RBCC	AFRL Air Vehicles Directorate, Control Sciences Division, Control Systems Development and Applications Branch
AFRL/RITB	AFRL Air Vehicles Directorate, Information Directorate, Advanced Computing Division, Computing Technology Applications Branch
AIAA	American Institute of Aeronautics and Astronautics
AJAX	Asynchronous JavaScript And XML
AMPI	Adaptive MPI
API	Application Programming Interface
ARL	Army Research Laboratory
ASIC	Application Specific Integrated Circuit
ATR	Automatic Target Recognition
BR&T	Boeing Research & Technology
BSCW	Basic Support for Cooperative Work
CDRL	Contract Data Requirements List
COTS	Commercial off-the-shelf
CMap	Concept Maps
CMS	Content Management System
CPS	Cyber-Physical Systems
CPU	Central Processing Unit
CSWG	Community Standards Working Group
DARPA	Defense Advanced Research Projects Agency
DESERT	Design Space Exploration Tool
DoD	Department of Defense
DSP	Digital Signal Processor
DVD	Digital Video Disc
EAR	Export Administration Regulations
FPGA	Field Programmable Gate Array

FCS	Future Combat System
FIR	Finite Impulse Response
GPP	General Purpose Processor
GPU	Graphics Processing Unit
HCSS	High Confidence Software and Systems
HTML	Hyper Text Markup Language
IDE	Integrated Development Environment
IHMC	Institute for Human and Machine Cognition
IP	Intellectual Property
IIR	Internet Routing Registry
ITAR	International Traffic in Arms Regulations
MCAR	Mixed Criticality Architecture Requirements
MIMD	Multiple Instruction stream, Multiple Data stream
MoBIES	Model-Based Integration of Embedded Software
MPI	Message Passing Interface
NITRD	Networking and Information Technology Research & Development
NRL	Naval Research Laboratory
NSF	National Science Foundation
OCP	Open Control Platform
OESC	Ontology Evolution Steering Committee
ONR	Office of Naval Research
OpenCL	Open Computing Language
OpenMP	Open Multi-Processing
OSD	Office of the Secretary of Defense
OWL	Web Ontology Language
PI	Principal Investigator
PoP	Period of Performance
PVTOL	Parallel Vector Tile Optimizing Library
R&D	Research & Development
RDBMS	Relational DataBase Management System
RDF	Resource Description Framework

RISC	Reduced Instruction Set Computer
RTOS	Real-Time Operating System
SBIR	Small Business Innovative Research
SDK	Software Developer Kit
SEC	Software Enabled Control
SEI	Software Engineering Institute
SOSCOE	System-of-Systems Common Operating Environment
SPP	Signal Processing Platform
SPRUCE	Systems Producibility Collaboration and Experimentation Environment
SSTC	Software Systems Technology Committee
SDTATIC	Software Development Tools And Technology Software Development Tools And Technology Information Clearinghouse
STTR	Small Business Technology Transfer
S3	Software Systems Stockroom
TWG	Technical Working Group
UAS	Unmanned Aircraft Systems
UAV	Unmanned Aerial Vehicle
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
VM	Virtual Machine
VSIPL	Vector, Signal, and Image Processing Library
V&V	Verification and Validation
WDCO	Boeing Washington DC Office
WG	Working Group
XML	Extensible Markup Language