Smart Aerospace eCommerce: Using Intelligent Agents in a NASA Mission Services Ordering Application[1]

Edward Luczak, Computer Sciences Corporation, 7700 Hubble Drive, Lanham, MD 20706
301-794-2388, eluczak@csc.com

Walter Moleski, NASA Goddard Space Flight Center, Greenbelt, MD 20771
301-286-7633, Walt.Moleski@gsfc.nasa.gov

Kim Morris, Honeywell Technology Solutions, Inc., Houston, TX
281-853-3237, Kim.Morris@csoconline.com

Bill Clayton, GHG Corporation, 1100 Hercules Ave, Houston, TX 77058
281-853-3169, Bill.Clayton@csoconline.com

Patricia Scherf, Lockheed Martin, Houston, TX
281-853-3134, Patricia.Scherf@csoconline.com

Abstract— This paper describes how intelligent agent technology was successfully prototyped and then deployed in a *smart eCommerce* application for NASA. An intelligent software agent called the *Intelligent Service Validation Agent (ISVA)* was added to an existing web-based ordering application to validate complex orders for spacecraft mission services. This integration of intelligent agent technology with conventional web technology satisfies an immediate NASA need to reduce manual order processing costs.

The ISVA agent checks orders for completeness, consistency, and correctness, and notifies users of detected problems. ISVA uses NASA business rules and a knowledge base of NASA services, and is implemented using the Java Expert System Shell (Jess), a fast rule-based inference engine.

The paper discusses the design of the agent and knowledge base, and the prototyping and deployment approach. It also discusses future directions and other applications, and discusses lessons-learned that may help other projects make their aerospace eCommerce applications smarter.

TABLE OF CONTENTS

1. INTRODUCTION

Intelligent software agents are receiving considerable attention throughout the Information Technology (IT) industry, and are now beginning to appear in aerospace applications. Intelligent agents can be used to make many eCommerce applications smarter, and may eventually become routine design components in a wide variety of e-business solutions. Many new aerospace software applications have some characteristics of eCommerce applications, and may benefit from the use of intelligent agents.

This paper describes how we prototyped and then deployed an intelligent agent that validates orders in a NASA-wide mission services ordering application called the SOMO Standard Services Ordering Tool (SSSOT). Spacecraft missions use the SSSOT to place orders with NASA's Space Operations Management Office (SOMO) for services such as mission planning, on-orbit flight operations, orbit and attitude determination, tracking and data network operations, and spacecraft data processing. The SSSOT is used to order hundreds of millions of dollars of services per year.

Users access the system using standard web browsers to select services from the SOMO service catalog, which contains over 500 standard services. Services must be ordered for each year of the mission lifetime; the mix of required services is typically different for each mission phase. Ordering services correctly requires extensive knowledge of the services offered and their relationships. Order errors occur frequently, and tedious manual checking has been required to catch them. Undetected errors could

result in costly resource planning problems. SOMO was expending significant effort in manually validating orders, and needed an approach to automate this validation task. Ideally, validation would be performed interactively when the user initially entered the order, and errors would be detected and corrected at the source.

Using expert system technology similar to that used to automate satellite control centers, we built an Intelligent Services Validation Agent (ISVA) that checks orders for completeness, consistency and correctness, and immediately notifies the user of detected problems. ISVA uses NASA business rules, decision aids, and a knowledge base of NASA services, and is implemented using the Java Expert System Shell (Jess). Jess is a rule-based inference engine developed at Sandia National Laboratory that is tightly integrated with the Java environment. The knowledge base defines an ontology of mission services, and describes ordering constraints between services and groups of services. The knowledge base was constructed with the help of the NASA/Contractor team of Customer Service Representatives who have detailed knowledge about mission services, and experience in helping missions order services.

The ISVA prototype was built on a Windows 2000 platform to demonstrate the concept, using the ColdFusion web application development tool and Microsoft's Access 2000 database. The production version of ISVA is deployed on a HP Unix platform with an Oracle database. Additional agents are planned to advise spacecraft missions on how to select the proper services.

ISVA was built as a collaborative effort between NASA and contractor staff at the NASA Goddard Space Flight Center in Maryland, and at the NASA Johnson Space Center in Houston.

This paper also discusses other potential uses of intelligent agents in aerospace-related eCommerce applications. The paper will conclude with a discussion of lessons learned that may be useful to other projects that want to make their aerospace eCommerce applications smarter.

2. Technology Approach

In existing eCommerce applications, orders are often validated at the client browser, or at the backend database. Because of the complexity of the NASA mission services ordering process, neither of these traditional validation approaches was appropriate.

At the client browser, JavaScript code is often used to perform field type and range checking of input forms. Java applets can also be used for this purpose in the browser window. However, these client-side approaches are limited to simple error-checking tasks that do not require the validation of completeness or consistency of sets of ordered products or services.

At the backend database, much more sophisticated error checking can be performed. However, NASA mission and data services are organized in hierarchies of groups and subgroups that are not naturally represented in relational databases. Moreover, because a wide variety of consistency, correctness, and completeness tests are required, stored procedures can be difficult to write for this purpose, and multiple queries can be cumbersome and expensive to perform.

A rule-based expert system is well-matched to this type of problem. An expert system consists of an inference engine, and a knowledge base containing a set of rules and a set of facts. Constructing an expert system application involves defining the rules and facts. Each rule has a set of conditions and actions. When the underlying inference engine determines that the conditions for a rule are satisfied by available data, it fires the rule - meaning that the rule's actions are performed. Inference engines that use the Rete algorithm can match conditions and fire rules very efficiently. In a complex validation problem, expert system rules can be used to represent constraints that must not be violated. For example, some services are mutually exclusive – when one member of a set of services is ordered in a given year, then no other members of the set can also be ordered. An expert system rule can be used to easily represent this constraint. When an order is checked that contains more than one service from a mutually exclusive set, the rule fires, and notifies the user of the problem.

The Goddard Space Flight Center has considerable experience in using forward chaining expert system applications to perform monitoring and control functions in spacecraft control centers [1], [2]. A principal tool that has been used is the C-Language Integrated Production System (CLIPS) [3], which was originally developed at NASA Johnson Space Center. CLIPS is a forward-chaining expert system language and shell; it uses the Rete pattern matching algorithm. CLIPS-based expert systems have been used operationally in almost a dozen control centers to monitor spacecraft health and safety, and have controlled NASA's X-Ray Timing Explorer (XTE) and Gamma Ray Observatory (GRO) spacecraft during routine "lights-out" operations periods – when the control center is completely unstaffed.

Sandia National Labs has recently developed a Java-based tool that implements much of the CLIPS language. Called the Java Expert System Shell (Jess) [4], it can run on any platform that supports a Java Virtual Machine, including Windows NT/2000 and HP Unix – which are the platforms that were used for prototyping and deploying our smart eCommerce system. Based on NASA's considerable experience and confidence in using expert systems built with the CLIPS language, and NASA's desire for a tool that provided close integration with the Java environment, Jess was chosen as the basis for the ISVA agent.

ISVA is considered to be an *intelligent agent* because it is designed to have many of the characteristics that are commonly used to describe the nature of intelligent agents [5]. It is intelligent because it can perform inferences in a complex data environment. It is an agent because is continually available to perform a service on behalf of a user in a network environment. It is a separately executing entity, with a specified role and responsibility in the larger system. Although it is not currently designed to learn, it dynamically obtains new information about mission services and their constraints from a knowledge base that can be updated while ISVA is operating. ISVA is an example of a knowledge-based software agent that can make web applications smarter.

Figure 1 summarizes the technology approach. A conventional eCommerce application, consisting of a web server, web application, and databases, is augmented with a set of "smart technology" components to yield a *Smart eCommerce Application*. The smart technology components include an intelligent agent and a product/service knowledge base. The intelligent agent consists of an inference engine (Jess), and a set of business rules for validating orders. The knowledge base consists of a set of facts that represent the properties, relationships, and constraints of the mission services offered by NASA.

3. CONCEPT PROTOTYPE

The concept of using an intelligent agent to perform mission service order validation was introduced to SOMO in July 2000, and was favorably received. However, because the technology was still unproven in this type of production environment, a concept prototype was planned to demonstrate the viability of the approach.

The objective of the concept prototype was to demonstrate the feasibility of using an intelligent agent to help users create NASA service orders more quickly, more completely, and with fewer errors. An ISVA concept prototype was designed to demonstrate the use of business rules and decision aids to validate orders in three ways:

- Consistency – check for incompatible services.
- Completeness – check for missing services.
- Correctness – verify that the proper services are ordered based on NASA guidelines and mission characteristics.

The concept prototype was also designed to demonstrate how ISVA could be integrated into the existing web-based ordering application, the SSSOT. The SSSOT was a new system in operation for less than a year, and major architectural changes to the system were undesirable. The SSSOT is implemented on HP Unix platforms, using the iPlanet web server, CGI Perl scripts, and a backend Oracle database. Because access to the operational SSSOT system was restricted, the ISVA concept prototype was designed as a standalone system, with a web user interface that emulated the operational SSSOT user interface. Windows NT and Windows 2000 Server platforms were available for developing the prototype, but no HP Unix computers. The prototype design therefore had to be easily ported from Windows to the HP Unix environment.

The tools used for the concept prototype were:

- Jess, the Java Expert System Shell - to provide the framework for the reasoning agent. Jess version 5.1 was used, with Java JDK version 1.2.2.
- ColdFusion – web development tool used to build the user interface and database interface.
- Microsoft Access 2000 – relational database used to store service order data.
- Microsoft Excel 2000 – spreadsheet used as a flat file database for storing the NASA Mission Set database.

Figure 2 shows a block diagram of the ISVA concept prototype system design. The prototype system was built in two major components that execute on two separate network-connected computers. The upper component, implemented on a Windows NT Server machine, emulated the operational SSSOT system. Microsoft Internet Information Server (IIS) was used as the web server, and ColdFusion was used to implement the user interface (UI) pages and the interfaces to the Access 2000 and Excel 2000 databases. ColdFusion is a parsed HTML web development tool that makes it very easy to create dynamic web user interfaces. It uses ODBC to connect to data sources such as Access and Excel, and allows database queries to be easily coded in SQL.

The lower component is the ISVA intelligent agent, hosted on a Windows 2000 machine. The agent is implemented with Jess, which loads a rule base when it is launched. The rule base then causes a Validation Knowledge Base of facts to be loaded. The rule base and Validation Knowledge Base are constructed and managed separately for the following reason. The rule base is coded in generic business rules that will change only infrequently, whereas the Validation Knowledge Base will be updated perhaps monthly, to reflect new services and service constraints.

The ColdFusion application and the ISVA agent communicate using a simple file protocol. For each validation transaction, the ColdFusion application dynamically prepares a file which contains the order data to be validated, and makes it available to ISVA. After validating the order, ISVA prepares a set of files that contain the validation results, and makes them available to the ColdFusion application. The exchange of these files is represented by the arrow connecting the ColdFusion application and ISVA in Figure 2. This interaction is described in more detail in the next paragraph.

When the user prepares an order, he or she can click a button on the order page to request that it be validated. When this happens, the ColdFusion application retrieves the order data and the characteristics of the user's spacecraft mission from the Service Request (SR) and Mission Set databases, writes the data as a set of facts into a text file, and notifies ISVA that a validation should be performed. ISVA reads this fact file, and then analyzes the order data and mission characteristic data using its business rules and the service definitions and constraints represented in the Validation Knowledge Base. ISVA detects any validation errors, and sends a text file report back to the ColdFusion application. ISVA also sends data that enables the ColdFusion application to highlight items on the order page using color-coding to indicate the severity of the validation error.

The ISVA concept prototype was demonstrated in September 2000. Even with a starter Validation Knowledge Base, the prototype was able to detect a wide range of order problems. Validations of typical orders required less than a second. The prototype was declared successful in demonstrating the concept, and funds were allocated to enable the prototype to be evolved to a production system. A second prototype was developed and demonstrated to the target user community in January 2001; feedback helped guide development of the production release.

4. Agent and Knowledge Base Design

Agent Design

The agent is designed as a set of generic *business rules*. Business rules describe the requirements and constraints on how NASA offers mission services. The latest release of ISVA contains 18 business rules. They are *generic* business rules because each business rule can represent many specific constraints. The specific constraints on how services may be ordered are represented by facts in the Validation Knowledge Base. The latest ISVA Validation Knowledge Base contains over 200 facts describing individual constraints.

An alternate design for a validation agent would be to represent each constraint in its own rule. This approach would lead to a large number of rules. One of the problems that many developers have experienced, however, is that expert systems having large numbers of rules become difficult to understand and maintain. Facts are much easier to maintain than rules. By using a few generic rules, ISVA is likely to be much easier to maintain.

ISVA's business rules are organized in a flat structure that reduces rule chaining. Although forward chaining is a powerful feature of expert systems, designs that involve extensive chaining can lead to systems that behave in ways that is difficult for others to understand. ISVA's flat rule design makes is easy to add new generic business rules.

An example of a generic business rule is:

If the Validation Knowledge Base specifies that
"Service X Requires Service Y"
and Service X is ordered, but Service Y is not ordered,
then a validation error is detected.

This business rule can be activated by any of a set of constraint facts that state: (Requires X Y), meaning that Service X requires Service Y. For example, the service "Attitude-Dependent Orbit Determination" requires the service "Attitude Determination." Other business rules deal with "prohibits" constraints, and constraints involving groups of services. For example, if any one of the services in the group "Orbit Determination Services" is ordered, then at least one of the services in the group "Tracking Services" must also be ordered.

Knowledge Base Design

The Validation Knowledge Base contains facts that describe the objects and relationships in the mission services domain. There are over 500 services in the SOMO services catalog, organized into a hierarchy of groups and subgroups. Some constraints apply to individual services, and other apply to groups of services.

Figure 3 illustrates the types of relationships that are represented in the Validation Knowledge Base. A major portion of the knowledge base is the definition of service groups and subgroups, which is a type of ontology of services. Individual services and service groups can be related by either "requires" or "prohibits" relationships in a variety of ways. Mission characteristics codes used in the Mission Set database have "allows" relationships with service groups. This relationship indicates which services are appropriate for various types of missions, or for various phases in a mission's lifetime. If a service is ordered that is not allowed by a Mission Set code for that mission, a validation error is detected. For example, launch services are not allowed to be ordered in a year other than when the launch occurs, and tracking services are not allowed for a mission that is only in the formulation stage.

The knowledge base contents were derived from knowledge engineering sessions conducted with NASA mission services experts. These engineers serve as Customer Service Representatives to mission project personnel, and help them select the appropriate services for their missions. A process was defined for the Director of Customer Services to formally verify and approve the contents of the Validation Knowledge Base, and its periodic updates.

5. PRODUCTION SYSTEM DEPLOYMENT

ISVA Release 1.0

The second prototype was incrementally evolved into the production release by adding functionality, increasing error handling robustness, and constructing the full Validation Knowledge Base. The overall ISVA design was not changed in moving from the prototype to the production environment. The production environment uses HP Unix servers, the iPlanet web server, a service ordering application programmed primarily in Perl, and an Oracle database. The file-based interface between ISVA and the ColdFusion application in the prototype was used without change for the production system. The production ordering application was modified to use this interface.

A month was allocated to determine and resolve problems that might arise in porting the ISVA agent software from the Windows to the HP Unix environment. However, when HP's version of the Java JDK 1.2.2 was installed on the HP Unix machines, the Java-based ISVA agent software ran exactly the same as on the Windows NT/2000 machines.

The ISVA prototype was developed and demonstrated in Maryland at NASA Goddard Space Flight Center, but the SSSOT production servers and development team reside in Houston at NASA Johnson Space Center. Travel money and time was budgeted for the Maryland team to visit Houston several times to participate in design discussions and support integration testing. However, by using several tools that can support distributed development teams, no travel was necessary – saving both time and money. The tools included:

- Microsoft NetMeeting – a point-to-point network conferencing tool that enabled the teams to give real-time prototype demonstrations and slide presentations to one another, just as if the two teams were in the same room.
- Xerox DocuShare – a web-based shared document library that was used for all project documentation and software deliveries.
- Teleconferences – traditional speakerphone-based teleconferences were used for weekly design and status meetings.

ISVA Release 1.0 became operational in April 2001, and has been in production use since that time. Users have been pleased with the new validation capability. Figure 4 illustrates the deployed system design.

ISVA Release 2.0

Development of ISVA Release 2.0 has been completed, and it is expected to become operational in early Fall 2001. This new release increases validation functionality by adding several new business rules, and also contains a variety of features that will enhance usability and ease maintenance. The execution environment was advanced to Java JDK version 1.3 and Jess version 5.2, the latest released versions of these tools. Of special interest in ISVA Release 2.0 are two new features that facilitate understanding and maintenance of the knowledge base: the knowledge base map, and knowledge base consistency checking.

Our early experience with ISVA operations indicated that updates to the knowledge base would be frequent, and somewhat tedious. The validation knowledge base is represented in a text file in CLIPS fact format. Updates will be made by selected Customer Service Representatives, and approved by the Director of Customer Service. Tools were needed to help understand the knowledge base, and to avoid errors when updates were made.

The knowledge base map is a web page generated automatically by the agent whenever it loads the current knowledge base. The map shows the structure of service groups and constraints, and uses hyperlinks to allow easy navigation between elements. The agent also checks the internal consistency of the knowledge base, ensuring that any service group or service referenced in a constraint is properly defined.

6. FUTURE DIRECTIONS AND OTHER APPLICATIONS

Future Directions

Several exploratory directions have been identified for future development of this smart aerospace eCommerce application. Two of these directions relate to management of the validation knowledge base; the third, expands the focus of the role of intelligent agents in the aerospace eCommerce domain.

We recognize the need to provide better tools for managing the Validation Knowledge Base – the set of facts that define the services, service groups, and constraints that the business rules use to perform validations. Better tools would make it easier to verify, approve, and update the knowledge base – and reduce the possibility of introducing errors when making updates. One approach we have begun to investigate is the use of a relational database to store these facts, with a simple web-based form interface that enables users to view, search, sort, and update the facts. We have built a prototype Knowledge Base Administration Tool using ColdFusion and Access 2000, and we are evaluating its utility. Another approach we are investigating is the use of a knowledge acquisition and knowledge base editing tool such as Protégé-2000 [6], which has been developed over several years by the Stanford Medical Informatics Group at Stanford University Medical School. This Java-based tool is enjoying widespread use in many fields.

We have also begun investigating the use of XML as a standard knowledge base fact representation syntax. The Validation Knowledge Base is currently represented in CLIPS fact format. Using a neutral, standards-based format that is independent of any particular inference engine would enable the use of a growing family of XML editing tools, and would facilitate the use of different inference engines, if desired.

In addition to validating orders, intelligent agents could also proactively advise customers about the products and services that they may need. We have begun to investigate the concept of using multiple Intelligent Service Advisor agents to advise users. Each agent would have expertise in a specific discipline, e.g., flight dynamics, or Space Network services. A multi-agent approach could make it easier to develop the needed knowledge bases – and also make it somewhat easier to acquire the knowledge from discipline experts.

Other Applications

Intelligent agents can be useful in other web-based aerospace ordering applications for both order validation, and for providing selection advice. A broad range of applications deals with ordering data products from science data archives. Huge archives of satellite-acquired Earth science, weather, and space science data are being connected to the web and made accessible to users.

For example, NOAA's innovative Satellite Active Archive [7] provides a rich variety of data products to the public via the web. Users order images and data products derived from polar-orbiting NOAA and DMSP satellites. Data products included sea surface temperature, climate measurements, cloud coverage, and snow and ice coverage. Orders may specify time range, geographical area, data type (and instrument), receiving station, satellite type, and other selection criteria. Intelligent agents could be used in a system like this to help users place realistic orders that better suit their needs, don't violate constraints, and avoid the delivery of extraneous unwanted products.

Intelligent agents can also be integrated with other types of aerospace web applications to make them smarter – including applications in resource scheduling, project and operations planning, and knowledge management.

7. Conclusions and Lessons Learned

Several conclusions and lessons-learned have resulted from this 15-month effort.

- ISVA has shown that an intelligent agent can be effectively added to an existing web application to make it smarter. There are at least some applications where this can be done without requiring the web application to be significantly reengineered.
- Building a rapid concept prototype is an effective way to initially introduce this new technology to an organization that is responsible for a fielding a production system. We used an iterative prototyping methodology to develop an initial rapid prototype, to extend it to a second more comprehensive prototype, and to evolve this into the production system.
- One of the classic problems in expert system development, that of ending up with an intractable bag of rules, can be avoided by designing the agent to with a small set of generic rules, and by putting most of the specific knowledge in facts.
- A simple data interface between the agent and the web application may be adequate, especially in applications with light transaction loads.
- The programmer's warning, "Java: Write once, debug everywhere," doesn't necessarily apply everywhere. Our Java-based Jess application ported effortlessly from Windows NT/2000 to HP Unix.
- A geographically distributed development team is feasible with traditional teleconferences strengthened with collaboration support tools, such as a web-based document library, and NetMeeting for remote slide presentations and demonstrations.

References

[1] W. David Ripley, III, Edward C. Luczak & Karl R. Mueller, "The Generic Spacecraft Analyst Assistant," *1998 CLIPS Conference*, proceedings published in [3].

[2] Jonathan Hartley, Ed Luczak & Doug Stump, "Spacecraft Control Center Automation Using the Generic Inferential Executor (Genie)," *Fourth International Symposium on Space Mission Operations and Ground Data Systems (SpaceOps 96),* Munich, September 16-20, 1996.

[3] J. Giarratano & G. Riley, Expert Systems: Principles and Programming, 3rd edition, PWS Publishing, 1998.

[4] Ernest J. Friedman-Hill, "Jess, The Java Expert System Shell, Version 5.2," Technical Report SAND98-8206 (revised), Sandia National Laboratories, Livermore, CA, 2001.

[5] Mark Watson, Intelligent Java Applications for the Internet and Internets, Morgan Kaufmann Publishers, 1997.

[6] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, & M. A. Musen, "Creating Semantic Web Contents with Protege-2000," *IEEE Intelligent Systems 16(2):60-71*, March/April, 2001.

[7] NOAA Satellite Active Archive, an online resource provided by the U.S. Department of Commerce, is accessible at http://www.saa.noaa. gov.

Ed Luczak *is a Senior Consulting Engineer at Computer Sciences Corporation, where he provides information technology research, development, and consulting services to NASA's Goddard Space Flight Center. He was the contractor task leader for the development of the smart technology components of ISVA. He has a BSEE from the University of Dayton, and an MS in Computer Science from the University of Maryland. His current interests include smart web applications, intelligent agents, knowledge based systems, and the semantic web.*

Walt Moleski *is a software engineer in the Advanced Architectures and Automation Branch at NASA Goddard Space Flight Center. He was the NASA Project Manager for the ISVA project. He has an MS in Computer Science from George Washington University, with a specialization in User Interfaces. His current interests include web applications, Java development, and user interfaces.*

Kim Morris *is a Manager with Honeywell Technology Solutions, Inc. for the Consolidated Space Operations Contract in Houston. He was the contractor Project Manager for the Services Catalog, SSSOT and the ISVA development. He has an MS in Business from the Johns Hopkins University with a concentration in MIS. His current interests include web based business and e-commerce applications.*

Bill Clayton *is a System Specialist at GHG Corporation. He is the Project Lead for the SSSOT development/ sustaining engineering project. He is a Texas CPA with an MS in Administration from George Washington University. His current interests include enterprise-wide systems and database integration using leading edge web and database technologies.*

Patricia Scherf *is a customer support analyst with Lockheed Martin for the Consolidated Space Operations Contract. She was a SSSOT administrator and contractor Project Lead for the ISVA project. She has a BBA from the University of Texas at Austin, and is pursuing an MBA from the University of Houston. Her current interests include database design and development, web development, and business systems automation.*

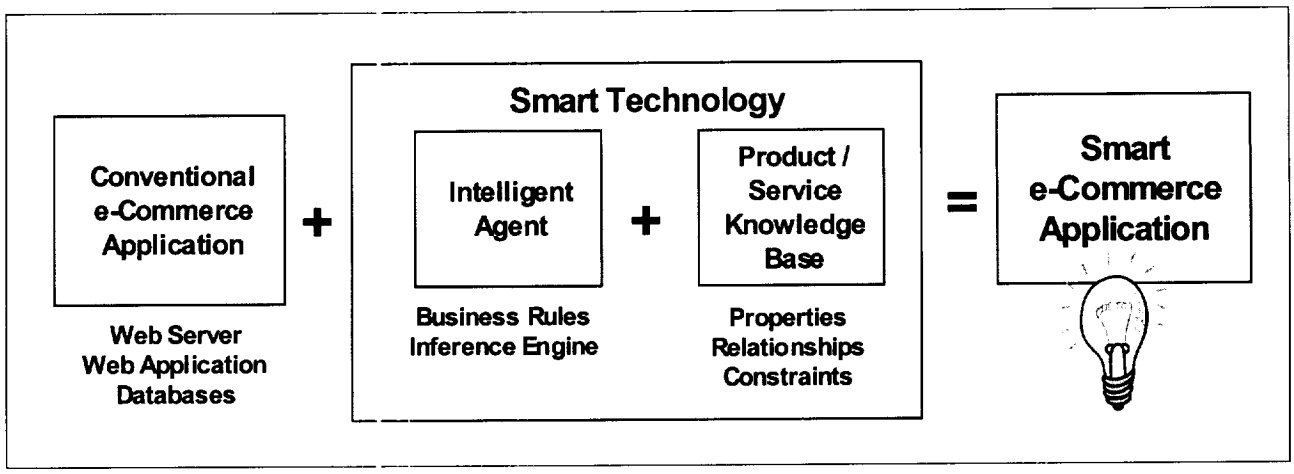

Figure 1. Technology Approach

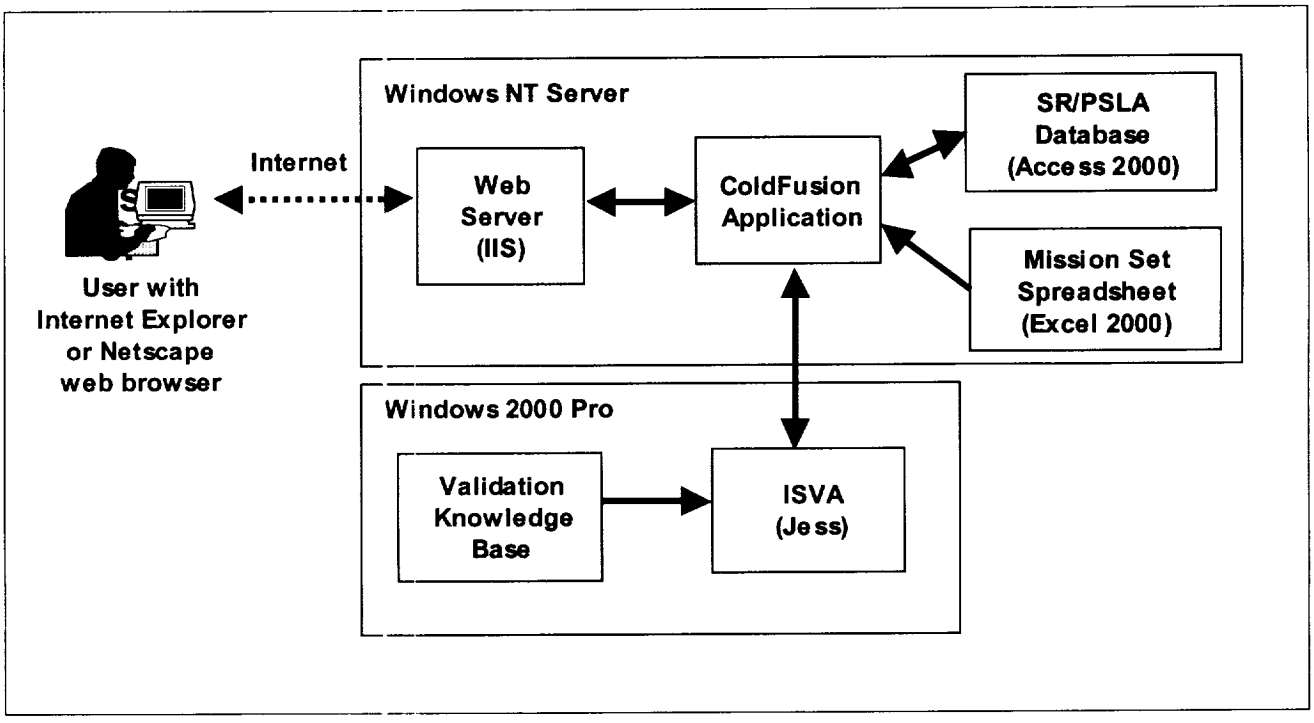

Figure 2. ISVA Concept Prototype System Design

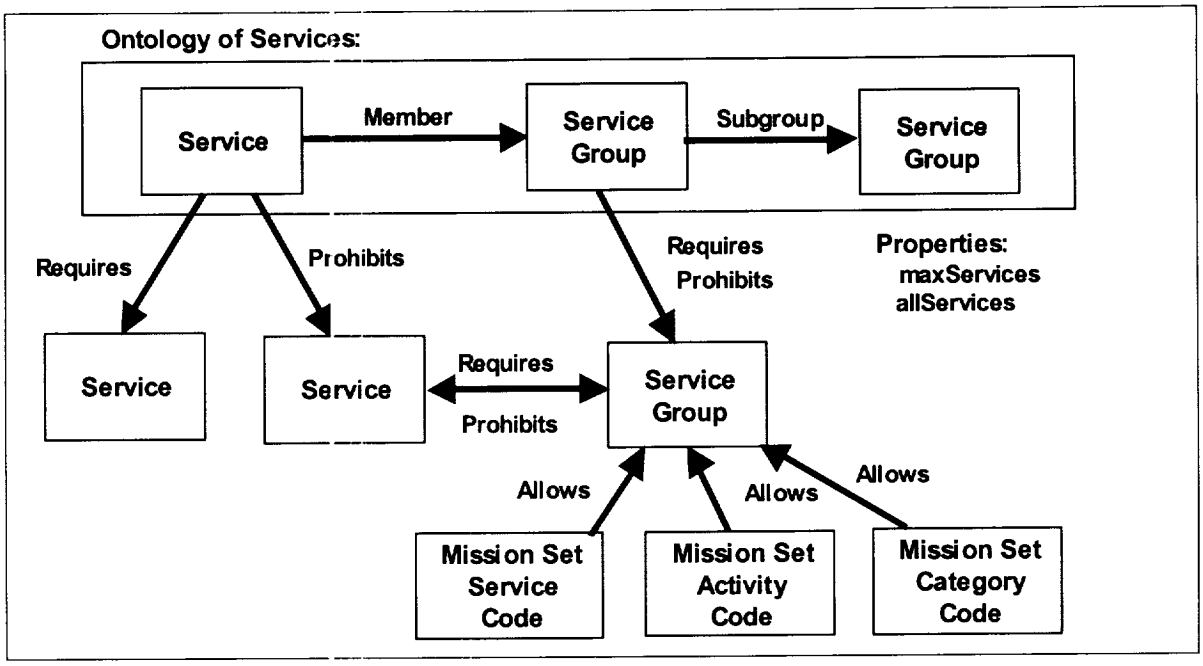

Figure 3. ISVA Validation Knowledge Base Design

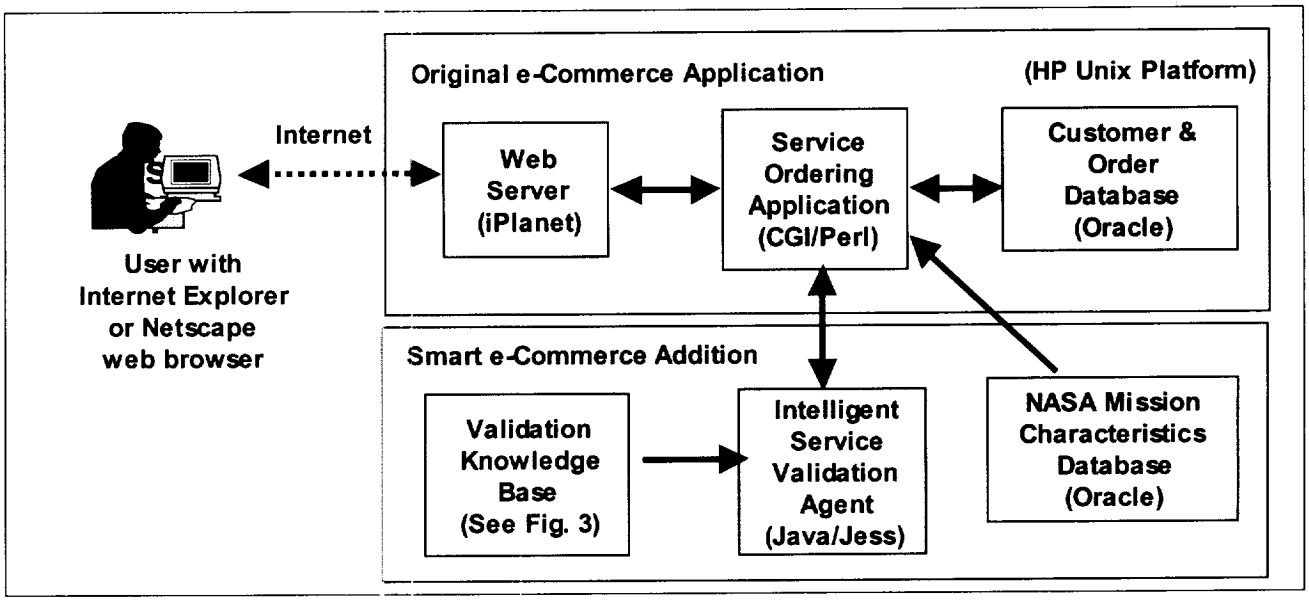

Figure 4. ISVA Smart eCommerce System Currently Operating at NASA's Johnson Space Center.