DEVELOPMENT OF A WEB BASED GROUP SUPPORT SYSTEM FOR LEADERSHIP EDUCATION

by

D. Nathan Hood

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Arts

Major: Teaching, Learning and Teacher Education

Under the Supervision of Professor David Brooks

Lincoln, Nebraska

May, 2008
DEVELOPMENT OF A WEB BASED GROUP SUPPORT SYSTEM FOR LEADERSHIP EDUCATION

D. Nathan Hood, M.A.

University of Nebraska, 2008

Adviser: David Brooks

The project described in this thesis involved the development of a Group Support System (GSS) for use in leadership education. The literature review is broken down into ten sections. First, virtual teams themselves are discussed, including some of the benefits and challenges posed by their use. Second, the concept of Adaptive Structuration Theory is discussed. Third, Media Richness is covered, followed by Media Stickiness. Fifth, the different technologies used by virtual teams are discussed, focusing on their advantages, disadvantages and effects on team processes. Next, Trust is covered, including how technology can influence trust formation, and how trust formation in turn affects group behaviors. The educational impacts of virtual teams are then discussed, with an emphasis on Collaborative Learning, and how information technology can affect, both positively and negatively, the learning process. Next, leadership is discussed. Finally, Group Support Systems and their effects on leadership are discussed. The thesis looks at the effects of technology upon leadership, the effects of leadership upon trust, and the effects of leadership upon the functioning of the group. After the literature review, the GSS is described, including providing a description of all functionalities implemented. The GSS
is a web-based group support system, designed for use in leadership education. The standard project management features are included (i.e. Projects, Goals, Milestones, and Deliverables). Additionally communication management functions are included in the GSS (i.e. Discussion Boards, and File Repositories). With the inclusion of the Discussion Boards, and other opportunities for group communication (Evaluation Criteria, Lessons Learned, and Facilitating Factors) the GSS has the additional flexibility for the different styles of leadership to be expressed by the group members allowing them to increase their knowledge about the processes of leadership. Figures showing the layouts used in the Group Support System are also included. Finally, the structure of the database used in the Group Support System is included as an appendix.
# Table of Contents

**Introduction** .................................................................................................................. 1  
**Literature Review** ........................................................................................................ 4  
  Virtual Teams .................................................................................................................. 4  
  Adaptive Structuration Theory ..................................................................................... 11  
  Media Richness Theory ............................................................................................... 13  
  Media Stickiness Theory ............................................................................................. 15  
  Virtual Team Technology ............................................................................................ 15  
  Team Tasks .................................................................................................................. 20  
  Trust ............................................................................................................................. 23  
  Education .................................................................................................................... 28  
  Leadership .................................................................................................................. 34  
  Group Support Systems .............................................................................................. 38  
**Recommendations for a Group Support System** ....................................................... 39  
**Project Description** .................................................................................................. 43  
  Site Presentation ......................................................................................................... 47  
  Site Functions ............................................................................................................ 56  
  Site Structure ............................................................................................................. 60  
**Discussion** .................................................................................................................. 61  
  User Feedback ........................................................................................................... 62  
  Limitations .................................................................................................................. 65  
**Appendix** .................................................................................................................... 67  
  Database Structure ..................................................................................................... 67  
  Stored Procedures ...................................................................................................... 116  
**Works Cited** ............................................................................................................... 181
Introduction

This project consisted of the construction of a website to be used by virtual teams. The reason for building this site was based on two objectives. First we wanted to be able to train leaders how to operate in a virtual environment. The Gallup Leadership Institute has done extensive research into the training of team leadership in the workplace, however comparatively little of this work has extended into the virtual team environment. To that end, the website is able to incorporate training materials in the form of videos and Power Point presentations available to the users to examine how individuals engage virtually using these tools.

The second reason for creating the website was to be able to investigate the operation of teams in the virtual environment and ascertain how leadership can be fostered in such an environment. This was accomplished by the creation of a very capable and flexible team environment where all of the normal project management functionalities are incorporated, and additionally, extensive functionality was added to facilitate interaction among the team members hopefully allowing for leadership to manifest itself in the virtual environment. Ideally, this will be able to tell us the type of team environment and what functionality is most useful for the effective operation of virtual teams, and inform us of how virtual teams can most effectively be lead.

Today people work predominantly in teams (DeSanctis & Poole, 1997; Malhotra & Majchrzak, 2005). In the past, these teams would all be geographically co-located and would work together face-to-face (F2F). As businesses become global entities, however,
it is more and more common for team members to be located in different locations leading to an increase in the prevalence of virtual teams.

The virtuality of a team is not an all or nothing proposition. A team’s virtuality can range from none at all in the case of an entirely co-located team, to extreme where none of the members of the team even work in the same country (Zigurs, 2003). Virtual team members can either work from their office with co-workers who are distributed either temporally or geographically, but a virtual team member can also be someone who teleworks, either from home or some off-site location.

There are numerous advantages to virtual teams in the modern workforce. First, they can bring more expertise to bear on a problem than a normal co-located team could by bringing the best minds from around the company to focus on the problem, rather than having to rely on the expertise to be found only in a single location (Zaccaro & Bader, 2003). Second, a company can see cost savings from virtual teams through, in the case of teleworkers, a reduction in the amount of office space required by employees as they work from outside the traditional office environment. Companies may also use virtual teams for recruiting purposes in cases where the company may be in a somehow unfavorable geographic location, or a prospective employee is reluctant to move (Cascio & Shurygailo, 2003). Third, by bringing a globally distributed group of people to bear on a problem, a greater level of cultural intelligence can exist within a team than when using team members from a single geographic location (Zaccaro & Bader, 2003).
By the end of 2008 a Gartner report predicted that virtual workgroups consisting of internal and contract employees will make up 60% of all offshore working arrangements (Bell, 2005). Employee surveys indicate that more employees would like to work remotely citing the flexibility it gives them (Cascio & Shurygailo., 2003). Thus, as time goes on, it seems likely that the use of virtual teams in the marketplace will increase.

The final outcome of this project will be a web site designed to assist the training of effective leadership behaviors in a virtual team environment. The web site will not only include the standard project management functions, it will also provide functions to encourage and enable communication amongst the team members. These additional functionalities should allow the different forms of leadership to manifest in the virtual environment, and allow team members to practice their leadership skills in a realistic environment.
Literature Review

Virtual Teams

In order for a web site that allows virtual teams to effectively interact, one needs to examine all of the different factors that go into the process of a virtual team working over the internet. The first factor that needs to be examined is definitional including, “what, exactly, is a virtual team?” A virtual team is a type of network form. A network form requires an existing communication infrastructure that facilitates the movement of information in a timely manner. Networks can operate (a) externally, working between a company and its suppliers for example; (b) internally, working between units within a company; or (c) both internally and externally. Network forms can either be stable or dynamic, but in either case assumes that the membership is participative and proactive and that the people working together do so largely out of choice (DeSanctis & Poole, 1997).

Network forms are already becoming apparent in today’s workplace as companies increase the number of joint ventures, increase their subcontracting and outsourcing activities even across international borders, and spin off new businesses from existing companies (Miles & Snow, 1986). Virtual teams have been defined as (a) groups of people that are either geographically or organizationally dispersed; (b) who accomplish their tasks by using a combination of communication and information technologies; and (c) rarely meet face-to-face. It is possible for virtual teams to be short-lived, existing only to accomplish a certain task, or they may be much longer lived existing to address
ongoing issues. In the longer lived teams, it is also possible for team membership to be highly fluid (Townsend, DeMarie & Hendrickson, 1998; Jarvenpaa & Leidner, 1999).

Lipnack and Stamps (1997) add that virtual team members interact through interdependent tasks and are guided by common purposes, which is the bedrock of all teams. Maznevski and Chudoba (2000) define global virtual teams as groups of people that have been identified by their organizations as belonging to a team, have responsibility for either making or implementing decisions about an organization’s global strategy, primarily communicate through technology rather than face-to-face and live and work in different countries. DeSanctis & Poole (1997) state that the team is the primary organizing structure in a networked organization, that the teams themselves will become the boundaries for collective work, and that team structures may ultimately replace the vertical structures that traditionally exist within an organization.

The virtuality of a team is not an all or nothing proposition. Several dimensions have been proposed in the literature that help to define a continuum along which the virtuality of a team may vary, even independently of the other dimensions. For instance, Kirkman, Rosen, Gibson, Tesluk and McPherson (2002) propose three dimensions along which the virtuality of a team may vary. The first is the proportion of time that team members work face-to-face as opposed to working virtually. The second dimension is the number of team members who work at a given location. The third dimension is how much time team members spend on their virtual team related tasks, rather than working on other tasks not associated with the team. Panteli & Davison (2005) expand upon the
dimension of geographical dispersion by including the concepts of cultural, ethnic and linguistic dispersion.

**Reasons for Virtual Teams**

To date, research into virtual teams has identified a wide variety of explanations for the trend of increasing virtual team usage. These include but are not limited to: access to skills or a larger number of workers (Townsend et al. 1998; Majchrzak, Malhotra, Stamps, Lipnack, 2004; Saunders, Van Slyke & Vogel, 2004), allowing workers to contribute to the organization, while remaining situated in their local support structures (Malhotra & Majchrzak, 2005), a more technologically literate workforce (Townsend et al. 1998), increased globalization of industries (Kraut, Steinfeld, Chan, Butler & Hoag, 1998; Townsend et al. 1998; Huysman, Steinfeld, Jang, David, Veld, Poot & Mulder, 2003; Jarvenpaa & Leidner, 1999; Jarvenpaa & Tanriverdi, 2003), increased flexibility in the organization, general industry trends (DeSanctis & Poole, 1997), an increasingly mobile workforce (Kayworth & Leidner, 2002), and the increased capabilities of modern technology (Jarvenpaa & Tanriverdi., 2003; Kirkman et al. 2002)

**The Future of Virtual Teams**

DeSanctis and Poole, (1994) have made some predictions about the future of virtual teams. Their first prediction is that as organizations move toward a network form, given a constant base of employees the number of teams within organizations will actually increase. The second is that teams will become more and more heterogeneous in their makeup as they draw from multiple levels within the organization and areas of
expertise. Their third prediction is that team structures will become more volatile in the future with teams quickly being formed and disbanded allowing for a greater flexibility in organizations. Fourth, they predict that teams will rely less on formalized procedures and more on information retrieval and sharing to accomplish their work. Finally, they predict that, as team members tasks evolve over time, so will the communications patterns within the team; this will lead to dynamic team processes and a reduction in routine.

**Benefits of Virtual Teams**

If an organization is looking to implement virtual teams into their structure the understanding of virtual team benefits and limitations is critical. One benefit to the deployment of virtual teams is improved access to skills (Malhotra, Majchrzak & Rosen, 2007; Zaccaro & Bader, 2003; DeSanctis, Fayard, Roach & Jiang, 2003), virtual teams provide improved access to skill by allowing for team membership to be drawn from a population that is not limited by being physically co-located, i.e. if an organization’s top design engineer is an Atlanta, but the rest of the group is in San Francisco, it becomes possible to include the engineer on the team. Another benefit of virtual teams is their asynchronous nature (Saunders et al. 2004; Avolio, Kahai, Dumdum & Sivasubramanian, 2001), where, with the use of computer technology to store files and keep track of discussions team members can work according to their own local time zones and not have to be concerned with working at the same time as the other team members to allow for proper coordination of the team’s work.
Virtual teams also convey benefits to the organizations that host them (Mowshowitz, 1997; DeSanctis & Poole., 1997; Kraut et al. 1998; Sole & Edmonson, 2002; Townsend et al. 1998; Huysman et al. 2003; Johnson, Suriya, Won Yoon, Berrett, & La Fleur, 2002; Miles & Snow, 1986; Cascio & Surygailo, 2003). For example, these benefits range from allowing companies to interact with acquiring inputs more efficiently to allowing for greater flexibility and potentially shortening product development and deployment cycles and allowing for more customization of goods and services.

There is also evidence that virtual teams can also improve communication within a team (Hamilton & Scandura, 2003; Townsend et al. 1998). By allowing for communication to be asynchronous virtual team members can function more flexibly as to when they can respond, which can stimulate additional connections between team members that may improve efficiency and quality of communication within the team.

Beyond allowing for greater flexibility, virtual teams can be structures that promote learning. For example, in an educational environment (Alavi, Yoo & Vogel, 1997; Johnson et al. 2002; Hiltz, 1998; Leidner & Jarvenpaa, 1995) virtual team interactions have been shown to increase student satisfaction possibly by allowing students to work at their own pace and allowing for students to participate in the learning process when they are most able. Such teams are also able to learn collaboratively in a virtual team environment which has shown to have benefits in student learning.

Some social benefits to virtual teams have also emerged (Johnson et al. 2002; DeSanctis & Poole, 1997). These benefits include people being less inhibited in the
virtual environment to offer ideas, and that as more lateral communication opportunities in a team environment arise, one might expect higher levels of identification with the team positioning the team to more effectively work together to achieve its goals.

**Limitations to Virtual Teams**

While virtual teams do have potential benefits to both organizations and people involved, there are also some potential limitations that have to be examined. The first limitation involves the nature of communication associated with virtual teams (Kayworth & Leidner, 2002; Häkkinen, 2004; Huysman et al. 2003; Robey, Khoo & Powers, 2000; Avolio, Kahai & Dodge, 2000; Kanawattanachai & Yoo, 2007). These potential communication limitations include but are not limited to: problems communicating information about the member’s context, problems communicating and understanding the importance or relevance of a piece of information, potentially uneven distribution of information, technical issues that may affect a member’s ability to access the information and problems interpreting silence from another team member.

Virtual teams may also introduce problems in team member motivation (Jarvenpaa & Leidner, 1998; Häkkinen, 2004; DeSanctis & Poole, 1997) where problems such as low individual commitment, absenteeism, the “Free-rider effect”, and the “sucker effect” can cause issues within the team environment hindering the team’s ability to work together. The lack of social and non-verbal cues can cause problems in the development of trust within the team (Avolio et al. 2001).
Getting the Most Out of Virtual Teams

Several researchers have suggested strategies for maximizing the benefits while minimizing the limitations of virtual teams. Kirkman et al. (2002) suggest that group communication should be monitored regularly, while also using rich communication media to provide performance related feedback to members. Also, training resources should be identified to allow develop team members to work in a virtual team environment in order to get up to speed faster. They also suggest the creation of a team mission statement, to be used in guiding the team’s processes and to help simplify distributed decision making.

Townsend et al. (1998) suggest using extensive scheduling functionality to keep all team members on the same page during the course of the team’s activities. Kreijns, Kirschner and Jochems (2003) suggest using social and non-task communication to build up community within the team to allow team members to work more effectively together. They also suggest five conditions that should be applied to aid collaboration within the team. First, is “positive interdependence,” meaning that team members should be linked together in such a way that each person’s success is tied to as many other people’s success as possible. Second, they suggest “promotive interaction” which encourages team members to help each other in reaching the team’s goals. Third is “individual accountability” where all group members are held accountable for doing their share of the work. Fourth, they suggest “interpersonal and small-group skills” which is the acquisition of skills to help team members operate in a team context. Finally is “group
processing” where the group itself determines which behaviors are appropriate or inappropriate based upon an examination of the group’s prior performance.

The prior section has reviewed general information about virtual teams, the benefits and limitations of virtual teams as well as best practices that can be implemented to get the most out of virtual teams. The next three sections will discuss various theories of importance in virtual team research: Adaptive Structuration Theory, Media Richness Theory and Media Stickiness Theory.

**Adaptive Structuration Theory**

To help explain one of the ways that teams behave in their environment DeSanctis & Poole (1994) expanded upon Structuration Theory and proposed what they called Adaptive Structuration Theory. In Adaptive Structuration Theory, a team takes structures or tools that exist in its environment and adapts them to their own needs through a process called appropriation. Structures are the specific set of rules, processes, resources and capabilities offered by a system. Structures can come from a variety of sources: appropriation of existing structures into new structures, the team’s environment and task, and the team internal system can all be sources of structure (Avolio et al. 2000). The source of the structure does not automatically dictate how the structure will be used by the team, in that the team actively chooses how it is going to use the structure being provided or appropriates it.

There are four aspects of the appropriation process that can be used to show the variety of appropriations a team can make: (a) directly use the structures as specified; (b)
the team can combine some structures with others creating a tool that behaves differently
than it did before; (c) limit or interpret how a given structure should be used, this also
modifies the way in which a given structure behaves; and (d) teams make judgments
about structures in their environment deciding whether or not the structure is something
useful to the team.

When a group appropriates a structure, the appropriation can either be faithful or
unfaithful. If an appropriation is faithful, the team has decided to use a structure in
accordance with its “spirit” or how the structure was intended by its designer or
originator to be used. In an unfaithful, sometimes called “ironic” appropriation (Boland,
Tenkasi & Te’eni, 1994), the structure is used against the intentions of the designer.
Unfaithful appropriations are not always due to misunderstandings on the part of the
team; they may also come from some realization of an unanticipated need whereby the
team uses a provided structure to “get by”, even if those appropriations may cause
redundancy or endanger security (Robey et al. 2000).

DeSanctis and Poole (1994) identify four factors that affect a team’s decision to
appropriate structures: the interaction style of the team’s members, the degree of
knowledge about and amount of experience with the structures embedded in the system,
the belief of the members that other members know and accept the use of the structures
and the extent of the agreement among team members about which structures should be
appropriated. Positive appropriation decisions by a team occur when their appropriation
patterns match the following properties: (a) appropriations are more frequently faithful
than unfaithful, (b) the team frequently appropriates structures, (c) the appropriations made are task oriented and for the benefit of the team rather than for the benefit of an individual member or purely for experimental purposes, and (d) the team has a positive attitude toward appropriations in general. How much impact the appropriation of a structure has on the performance of a team is determined by three factors: the structural potential of the structure being appropriated (structural features and capabilities), how technology and other structures have already been appropriated by the team, and new social structures that evolve over time within the team.

**Media Richness Theory**

Media richness is a theory advanced by Daft & Lengel (1986) to explain the differences in different formats of communication. According to Daft and Lengel, the richness of a medium varies because of the medium’s capacity for immediate feedback, the number of cues and channels utilized (visual, verbal, non-verbal), how personalizable the medium is and the variety of language supported by the medium. Face-to-face communication is the richest medium with simple numeric documents being the least rich (or leanest) medium.

Two important concepts in media richness theory are uncertainty and equivocality. Uncertainty is the absence of information that is required to make a decision, and is reduced by obtaining additional information. Equivocality is the existence of multiple, potentially conflicting interpretations of a piece of information. Organizations are designed typically to process knowledge in a variety of ways with the
ultimate goal of reducing both uncertainty and equivocality. The more information that can be carried in a message, the further it can go to reduce both uncertainty and equivocality. The richness of information is defined by the information’s ability to effect change in understanding within a specific time interval. If a piece of information can quickly cut across different frames of reference, or succinctly clarify ambiguous issues to change an individual’s understanding, then it is considered rich. If, however, a piece of information takes a long time to convey enough information to change understanding, or cannot overcome different perspectives, it is considered lean.

Since face-to-face communication conveys the most information (body language, tone of voice, amount of eye contact, physical attributes of the person sending the message etc.) it is considered to be the richest communication medium, while a simple sheet of numbers (a financial report, for example) is the least rich since it does not convey any information other than what is contained within. Interactions that occur with most individuals through computer mediated technology tend to be relatively lean. Tone of language can be conveyed, however eye contact, non-verbal cues and any physical information about the sender of the message is lost. This presents an enormous challenge to teams when trying to process information. Often times teams have found it necessary to go to richer media like teleconferences in order to effectively resolve disputes or process complicated information (Daft & Lengel, 1986; Järvelä & Häkkinen (2002).
**Media Stickiness Theory**

Media stickiness is a phenomenon by which the decisions made by a team about the use of technology early in its life cycle can constrain later flexibility in terms of technology use later in the team’s life cycle. Huysman et al. (2003) suggested that each team had a distinct approach to technology and developed their own habits of usage, and that these habits of usage were developed very early in the lifecycle of the team but continued throughout the team’s existence. One of the implications of media stickiness is that any interventions aimed at improving or altering the team’s use of technology needs to occur near the beginning of the team’s life cycle. An extreme example of this observed by Huysman et al. was in the case of a teleconference meeting in which the team was showing a diagram to the other group by holding it in front of the camera instead of taking a snapshot of it in Netmeeting and sharing it that way. The students were all very enthusiastic about the taking of snapshots, but afterward reverted back to process of holding diagrams in front of the camera.

The previous three sections discussed theories related to the various technologies implemented in a virtual team environment. The next section will discuss the different forms virtual team technology can assume.

**Virtual Team Technology**

Virtual team technology has been called a variety of names in the research: GDSS (Group Decision Support System), GSS (Group Support System), CSCW (Computer Supported Collaborative Work) and CMC (Computer Mediated Communication). In
terms of education the terms CSCL (Computer Supported Collaborative Learning), TML (Technology Mediated Learning) among others have been used. According to Zigurs and Buckland (1998), group support systems are systems that combine communication, computer and decision technologies together to support the formulation and solution of problems during group meetings. These authors have identified three main components to a GSS: support for communications, including input, display and feedback capabilities; support for process structuring, including enforcing agendas and other project management related functions; and support for information processing, including information storage, manipulation and structuring.

DeSanctis and Poole (1997) observe that since team members are dispersed and mobile, usage of electronic communication methods will become much more frequent, and that since these communication needs will change frequently, virtual teams will need to reply on multiple media options to facilitate coordination with people both inside and outside of the team. Virtual teams interact with each other through technology both synchronously (as in chat sessions or video conferencing) and asynchronously (as in e-mail or bulletin boards) (Avolio et al. 2001). Examples of these virtual team technologies include video and audio conferencing, shared calendaring, digital document management systems, text based group support systems, and more (Davison & De Vreede 2001).

Virtual team technology has several advantages for virtual team interactions. The software in these technology systems will facilitate the creation of group document and
presentations, aid in the analysis of data and assist with team level planning and decision making (Townsend et al. 1998). Assudani (2006) observed that technology allows virtual team members to communicate their contextual conditions to the rest of the group, and aids in the bridging of the geographic distance. Rich synchronous communication supports information sharing among team members by allowing rapid feedback assisting in issue clarification and allowing all team members to ensure they are referring to the same part of a document. The record keeping aspects of virtual team technology allow team members to understand the context of the information being generated or used by the team (Malhotra & Majchrzak, 2005). Contextualization of information is enabled by functions where team members can add comments or link additional pieces of information to a document (Boland et al. 1994).

When used properly the document management features of virtual team technology enable teams to maintain access to the most recent version of a document, and provides a place for all information relevant to the team to be collected (Rains & Scott 2006). Virtual team technology can also be an equalizer among members of a group. By removing the non-verbal cues and any indications as to status team members are placed on a more equal footing (Zigurs, 2003). Along with all of these benefits to virtual team technology, it is also necessary to be cognizant of its limitations. Many of the problems associated with virtual team technology center around the relative leanness of the communication methods allowed. While in some cases (such as videoconferences) many of the benefits of face-to-face communication are realized but not all of them are
Townsend et al. 1998). Virtual team technology use has been shown to shift team communications more toward process and task oriented messages and away from the social messages that give team members facilitate interactions between the members of the team (Robey et al. 2000).

While in some ways the asynchronous nature of communication within a virtual team can be a benefit, there are times where it has complicated the process of creating a common mental map amongst the team members (Järvelä & Häkkinen, 2002). (Please see Table 1 for a listing of the advantages and disadvantages of many different kinds of virtual team technologies.)

One of the challenges to communication in an online environment is the concept of grounding (Brennan, 1998). Grounding theory states that communication does not occur simply when a speaker produces an utterance; the speaker has to realize that his utterance has been accepted and understood as he intended. The challenge in a virtual environment often comes from the lack of feedback provided by both asynchronous communication and the tools themselves used to communicate.

People consider the most important form of communication in the online environment that of communication between the members of the team. Equally important, however, is the communication between the team members and the tools they are using.
Table 1 Key advantages and disadvantages of new communication technologies available to virtual student teams (Taken from Rains and Scott, 2006).

<table>
<thead>
<tr>
<th>Technology</th>
<th>Key Advantages</th>
<th>Key Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asynchronous discussion tools:</td>
<td>• Members do not have to be online at the same time</td>
<td>• Do not work well for intensive discussions</td>
</tr>
<tr>
<td>list-servs, discussion boards,</td>
<td>• Tools can complement face-to-face interactions</td>
<td>• Takes time – may not be useful when nearing a deadline</td>
</tr>
<tr>
<td>e-mail</td>
<td>• May provide a record of the interaction</td>
<td>• Some uneven participation</td>
</tr>
<tr>
<td>Synchronous discussion tools:</td>
<td>• Useful tool for brainstorming</td>
<td>• Requires everyone meet online at the same time</td>
</tr>
<tr>
<td>audio-conferencing, chat, video</td>
<td>• Can engage in an in-depth discussion</td>
<td>• All members must have access</td>
</tr>
<tr>
<td>conferencing, instant messaging</td>
<td>• Provides a record of the interaction</td>
<td>• Cost for video-conferencing tools</td>
</tr>
<tr>
<td>Coordination/collaboration groupware tools:</td>
<td>• Can aid coordination with calendar</td>
<td>• Must be maintained and kept up to date</td>
</tr>
<tr>
<td>Blackboard, SmartGroups, HotOffice</td>
<td>• Can aid collaboration via file sharing, shared documents, and tracked changes</td>
<td>• Price of commercial tools</td>
</tr>
<tr>
<td></td>
<td>• May serve a symbolic/identity function for team</td>
<td>• Privacy may be lost or diminished</td>
</tr>
<tr>
<td>Electronic meeting systems</td>
<td>• Adds structure to discussions, brainstorming and decision making</td>
<td>• Access and financial cost</td>
</tr>
<tr>
<td></td>
<td>• Creates opportunities for all members to contribute to discussion and decisions</td>
<td>• Requires a trained facilitator/technographer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Uneven acceptance of decisions</td>
</tr>
</tbody>
</table>
If the tools cannot provide adequate feedback about whether a command has been received and understood, then the tools will not be effectively used by the teams. This lack of effective use may lead to or mask problems in other team processes, not just how they use the tools (Rutkowski, Vogel, Genuchten, Bemelmans & Favier, 2002). Feedback from the virtual team technology is not simply an indication that a command has been accepted and understood, it also can be used to indicate what actions are available to the user. This form of feedback is called an affordance (Norman, 1990).

For virtual team technology, another important aspect of grounding is “forcing” the process of making it very difficult for an error to be committed by a user. Unfortunately, if forcing is used too often, as in the case of Windows Vista constantly asking one to confirm actions, the user will become frustrated and either automatically clicks yes even when they didn’t mean to or find a way to disable the forcing feature.

**Team Tasks**

Any examination of the technology used by virtual teams needs to consider the types of tasks that the team will be asked to perform, a concept called task fit (Zigurs & Buckland 1998, Kayworth & Leidner, 2002). Zigurs and Buckland (1998) use four dimensions to measure the complexity of a task. The first is “outcome multiplicity” which is simply the number of things the team is trying to accomplish. The second is “solution multiplicity” which is the number of possible solutions to the problem; each may have its own advantages and disadvantages. Third is “conflicting interdependence” which is that fact that it is possible for a decision to pursue a possible solution early in the
team’s process may preclude them from choosing another possible solution in the event
of a problem. Last is “solution scheme/outcome uncertainty” which measures how
certain the team is that the solution they have proposed will fix the problem they have
been tasked with solving. Zigurs and Buckland (1998) go on to define five different
types of tasks that vary on the four dimensions stated above (see table 2).

Table 2 Aggregated Task Categories (Adapted from Campbell 1998)

<table>
<thead>
<tr>
<th></th>
<th>Simple Tasks</th>
<th>Problem Tasks</th>
<th>Decision Tasks</th>
<th>Judgment Tasks</th>
<th>Fuzzy Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outcome Multiplicity</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Solution Scheme</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Multiplicity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflicting</td>
<td>No</td>
<td>Yes or no</td>
<td>Yes or no</td>
<td>Yes or no</td>
<td>Yes or no</td>
</tr>
<tr>
<td>Interdependence</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solution</td>
<td>Not applicable</td>
<td>Low to high</td>
<td>Low to high</td>
<td>Low to high</td>
<td>Low to high</td>
</tr>
<tr>
<td>Scheme/Outcome</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncertainty</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The authors also break down the functionality in a group support system into
three main dimensions. The first is communication support, which can be defined as any
aspect of the technology that enables group members to communicate with each other.
This can take the form of either group or individual messaging. Second is process
structuring, which affects or controls the way in which groups interact, from agenda
setting to creating a record of the group’s activities. Finally is information processing,
which enables team members to gather, share, evaluate and organize information.
Table 3 displays the five types of tasks in table 2, and displays which of the group support system technology dimensions is the best fit for work on that type of task.

Table 3 Fit Profiles of Task Categories and Technology Dimensions (Taken from Zigurs and Buckland, 1998)

<table>
<thead>
<tr>
<th>Task Category</th>
<th>Communication Support Dimension</th>
<th>Process Structuring Dimension</th>
<th>Information Processing Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Tasks</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Problem Tasks</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Decision Tasks</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Judgment Tasks</td>
<td>High</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Fuzzy Tasks</td>
<td>High</td>
<td>Medium</td>
<td>High</td>
</tr>
</tbody>
</table>

So, in the case of a decision task, the virtual team would select a virtual team technology that provides low communication support, high process structuring support and high information processing support.

In order for virtual teams to effectively use the technology they have been provided, they have to ensure that the task they are attempting is adequately supported by the technologies they will be using. The virtual teams literature also lists other suggestions for virtual team members looking to make best use of the technology they have been provided. Malhotra et al. (2007) provide some guidelines that can be followed to enhance technology usage. The first is that the time between meetings should be used for asynchronous communication to generate and evaluate ideas. Using this strategy
allows team members to contribute at times that are convenient for them, and gives them more time to formulate their ideas. Also identified was developing a sense of experimentation among members in using technology that can aid in further developing the team. This developmental strategy can prevent frustration with failures in the technology as they may have found other technologies that work, or have found a way to work around the problem.

Rains and Scott (2006) identified the use of online calendaring as an important tool that a team can use to improve performance. Using this tool becomes important in two ways: it can be used to keep the whole team apprised of the team’s progress towards their goals, but it can also be used as a form of contract among the team members where the team’s expectations in terms of project or deliverable completion dates are made explicit.

**Trust**

In an environment like that of a virtual team where it is impossible to closely monitor the actions of your teammates, trust is a critical factor in team performance (Avolio et al. 2001). Jarvenpaa and Tanriverdi (2003) identify three reasons why trust is so important to a virtual team’s development and performance. First, is that virtual teams are rife with uncertainty and can be prone to conflict. Second, virtual teams have a multitude of stakeholders, each with their own set of motivations. In this potential conflict-rich environment trust is critical to helping to avoid or manage conflicts and settle disputes. Finally virtual teams are inherently limited by their technology; the
traditional process of building trust through social conversations or personal interactions, are more difficult in a virtual team environment.

Trust between individuals is defined as “the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party” (Mayer, Davis & Schoorman, 1995. p712). In virtual teams, trust is linked to teammates perceived ability, benevolence and integrity (Jarvenpaa et al. 1998). Ability refers to the skills and aptitudes that allow an individual to be seen a capable and competent by their teammates (Jarvenpaa et al. 1998; Mayer et al. 1995). Benevolence refers to the extent to which a teammate is believed to be willing to help out their teammates for the benefit of the team, rather than their own personal gain. Integrity is the extent to which an individual is seen as adhering to a set of principles that should make them dependable and reliable, and that they will act over time in a manner consistent with these beliefs (Jarvenpaa et al. 1998; Mayer et al. 1995).

Conditional trust is one of the first types of trust developed by virtual team members (Jarevenpaa et al. 1998). Conditional trust requires that group expectations are laid out clearly and can stem from the knowledge of a team member’s education and experience (Avolio et al. 2001). Conditional trust in a virtual team has been shown to follow what has also been referred to as the swift trust model at times (Jarevenpaa et al. 1998). In swift trust, members import trust and start off trusting each other based on things such as perceived competence or roles served by team members. As time goes on,
the team member’s actions will determine if swift trust develops into other forms of
deep trust that would be more based on relationships (Mayerson, Wieck & Kramer,
1996).

Kirkman et al. (2002) believe that team member trust can be formed without face-
to-face communication. In contrast, Handy (1995) states that “trust needs touch.” In
normal face-to-face interaction, trust develops through the maintenance of confidences
and social interactions around a water cooler or after work gatherings; this is a form of
interpersonal or benevolent trust. In the virtual team environment, the social interactions
to form this kind of trust are not present. Instead, trust develops from people following
through on their obligations consistently and being responsive to requests from others;
this is a form of ability-based or task-based trust.

**Increasing Trust in a Virtual Team**

Since trust has been shown to be so critical to team performance, it is important to
review what can be done to increase trust within a virtual team. Different aspects of a
team’s organization can serve to increase trust among its members. For example, a team
that has consistent membership and opportunities to work together increases member
familiarity, potentially increasing levels of trust among team members (Lipnack &
Stamps, 1997).

Alternatively, actions and behaviors on the part of team members can serve to
increase trust. The following behaviors have all been shown to increase trust on the part
of a team: frequent task oriented communication (Kanawattanachai & Yoo, 2007;
Jarvenpaa & Leidner, 1999), demonstrating commitment to the team (Vogel, Genuchten, Lou, Verveen, Ven Eekout & Adams, 2001), showing initiative and enthusiasm for the work of the team (Jarvenpaa & Leidner, 1999), providing others with substantive feedback (Javenpaa & Leidner, 1999; Kirkman et al. 2002), and demonstrating one’s reliability to the team (Malhotra et al. 2007; Orvis & Lassiter, 2006).

Virtual team technology also has the ability to increase the trust within a team. In rich media environments, technology can assist in the formation of trust among homogeneous groups by assisting in the formation of social identification (Daft & Lengel, 1986). In leaner media environments, technology assists in the formation of trust by promoting the sharing of task-relevant knowledge. Lean media environments present some advantages for the development of trust over richer media environments. They can reduce the cognitive load that needs to be processed by team members thus allowing them to concentrate more on task oriented messages. Lean environments can also filter away many of the non-verbal cues (appearance, accent) that may lead to separation among the teammates and facilitate a sense of “sameness” among the team members increasing the level of trust felt (Avolio et al. 2001).

**Decreasing Trust in a Virtual Team**

Equally important to team designs as team member behaviors and aspects of technology that increase trust are those that can decrease trust. In terms of team structure, the main thing that can decrease trust within a team is the presence of subgroups (Panteli & Davison, 2005). Jarvenpaa and Leidner (1999) identified several
behaviors on the part of team members that can reduce trust within a team. These include: lapses in communication, complaints to administrators about other members, lack of participation on the part of team members, and teams not accurately envisioning what needed to be done to complete the team’s tasks.

Technology is also capable of decreasing trust. Majchrzak et al. (2004) observed that one to one exchanges amongst team members can lead to information asymmetry in the teams which can lead to distrust. However, copying everyone on an e-mail is not a solution to this, since it can lead to information overload on the part of the team members. One might be inclined to think that the record keeping abilities of virtual team technology can also aid trust building. While this is true, record keeping and behavior control technologies can actually lead to a loss of trust on the part of team members. Record keeping and behavior control technologies can reduce trust on a team by making problematic behavior on the part of a team member or team members more salient to the rest of the team (Piccoli & Ives, 2003). This would be similar to a case where a member of a team accesses logs showing that another team member did not access required files until mere minutes before a meeting (Huysman et al. 2003).

**Style of Communication and Trust**

Jarvenpaa et al (1998) found in their study that high trust teams communicated in a much more positive manner than the low trust teams. In the high trust teams, a disagreeing team member would propose an alternate solution rather than simply stating their disagreement. Additionally, low trust teams would focus on what might be lost due
to low performance, while high trust teams focused on what they stood to gain by superior performance. See Table 4 for a listing of strategies that may differ between high and low trust teams.

**Table 4 Strategies between High- and Low-Trust Teams (Taken from Javenpaa et al. 1998)**

<table>
<thead>
<tr>
<th>Behaviors/strategies</th>
<th>High-trust teams</th>
<th>Low-trust teams</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Style of action</td>
<td>• Proactive</td>
<td>• Reactive</td>
</tr>
<tr>
<td>2. Focus of dialog</td>
<td>• Task output driven</td>
<td>• Procedural</td>
</tr>
<tr>
<td>3. Team spirit</td>
<td>• Optimistic</td>
<td>• Pessimistic</td>
</tr>
<tr>
<td>4. Leadership</td>
<td>• Dynamic</td>
<td>• Static</td>
</tr>
<tr>
<td>5. Task goal clarity</td>
<td>• Team’s responsibility</td>
<td>• Individual responsibility</td>
</tr>
<tr>
<td>6. Role division and specificity</td>
<td>• Emergent and interdependent</td>
<td>• Assigned, independent</td>
</tr>
<tr>
<td>7. Time management</td>
<td>• Explicit/process-based</td>
<td>• Nonexistence</td>
</tr>
<tr>
<td>8. Pattern of interaction</td>
<td>• Frequent, few gaps</td>
<td>• Infrequent, gaps</td>
</tr>
<tr>
<td>9. Nature of feedback</td>
<td>• Predictable, substantive</td>
<td>• Unpredictable, nonsubstantive</td>
</tr>
</tbody>
</table>

**Education**

According to Leidner & Jarvenpaa (1995) business schools are experiencing increased competitive pressures, and they suggest using information technology as one way for a school to differentiate itself from others and attract students. Online learning has been defined as a socio-technical process. The technical aspects allow dispersed
learners with the proper tools to access information and knowledge spread across many different physical locations. The social aspects stem from gaining familiarity with others in distant locations to generate new learning (Assudani, 2006). However, according to Kreijns et al. 2003 the social aspects are often downplayed in online learning, and can cause problems in terms of learning in an online environment.

Koschman (1994) observed that most of the research to date done on learning in the virtual environment has not been done with much grounding in any form of learning theory. Instead, it has been done by focusing on the technology’s feature set, design, or technological procedures (Järvelä & Häkkinen, 2002). Hiltz (1998) made several observations about students in her online learning course. First she found that 65% of the students felt that taking courses online was more convenient than taking them in a traditional classroom environment. Next the students reported increased interest, involvement and motivation in the class working online. Amongst those who regularly participated in class, 55% agreed that the fact they knew their comments would be read and evaluated by others in the class increased their motivation, 62% felt that the class was less boring than traditionally taught classes, and 56% felt like they were able to take a more active role in the course. Students also felt that the class was more interactive, with 58% saying that they had better access to their instructor through the online course studied than in a traditional classroom. 47% of the student felt that in an online environment they had more communication with other members of the class, 19% saw no
difference in intra-student communication, and 32% felt there was more intra-student communication in a traditional class.

**Encouraging Learning in a Virtual Team Environment**

Leidner and Jarvenpaa (1995), building on the work of Nonaka (1994), identified three conditions that should exist to allow creative thinking through discussion in a team environment to allow learning. The logic used in the arguments needs to come from many sources, not just a single source; learners should feel free to express ideas openly and candidly; and the process should not be interrupted therefore affecting knowledge creation. This means that an effective team learning environment should allow for multiple forms of communication, some of them anonymous and some of them continuous.

Nonaka (1994) also observed that for learning to take place in an online team environment that without some form of shared experience, it is very difficult for team members to come together and share thought processes as a way of learning. As an advantage of learning in a team environment, Littleton and Whitelock (2004) state that computer technology allows students to have learning conversations at a distance with other students allowing them to construct knowledge through discussion with their tutors and peers.
Media Stickiness and Learning

In an online team environment, two types of learning have been identified by Huysman et al. (2003), “learning what” and “learning how.” “Learning what” refers to student’s learning about the project; “learning how” refers to the students learning how to communicate and interact in the online environment. In line with Media Stickiness Theory, they observed that groups stopped learning new ways to communicate and interact after the first one or two meetings, but did continue to learn about their project or task. Rains and Scott (2006) predict that the increase in the number of online courses and virtual teams in various organizational settings suggests that virtual student teams will become more common and even necessary in the future to prepare students for the online work environment.

Collaborative Learning in the Virtual Environment

Leidner and Jarvenpaa (1995) observe that information technology can enable the effective implementation of constructive, cognitive, collaborative and sociocultural models of learning. The virtual site, by encouraging communication and interaction of team members, primarily encourages the collaborative model of learning. According to Hiltz (1998), collaborative learning can be identified by the fact that both teachers and learners are active participants in the learning process, and that knowledge is not something given by the teacher to the students, but is something that emerges from the discussions and activities of the participants involved.
In order for learning to be considered collaborative, four conditions must be satisfied: resources available to the team must be shared, team members must be able to communicate ideas from one to another, attention has to be able to be focused and maintained by all participants on a similar part of the problem, and all participants must share in the successes and failures (Singer, Behrend & Roschelle, 1998).

Social interactions have been shown to be critical to the process of collaboration (Kreijns et al. 2003); however social interactions aren’t always positive and may affect the learning of the team negatively (Singer et al. 1998). To achieve effective collaboration, a number of social preconditions must be met: there must be shared understanding on the part of the participants, there has to be trust amongst the participants, and there has to be accountability (Kirschner & Van Bruggen, 2004). Singer et al (1998) use the term Notational System to describe how information technology behaves in a collaborative learning environment. A Notational System has three interrelated functions: mediating communication, support of activities, and encouraging productive learning.

According to Hamilton & Scandura (2003), high degrees of on-line interaction and extensive use of collaborative tools lead to higher levels of learning as reflected through better test scores for on-line learners when compared with those of students who learned in a traditional classroom. That is not to say that collaborative learning is not without its limitations. Leidner and Jarvenpaa (1995) observed that students may have trouble adapting to the lack of firm grading guidelines that can exist in a collaborative
environment and may have trouble accepting the additional responsibility that is placed upon them in a collaborative environment.

Kreijns et al. (2003) observe that the shift from traditional learning to online collaborative learning has fallen afoul of two pitfalls. The first is that it is taken for granted by educators and technology designers that just because the technology allows students to interact they will actually use it to do so in the manner intended by the designers, which as has been stated by Adaptive Structuration Theory, is far from a guaranteed occurrence. The second is that research has usually been done to work on the cognitive aspects of collaborative tools while the social or emotional aspects of those tools have been ignored.

Orvis & Lassiter (2006) have provided guidelines on how some of these limitations can be overcome. First, they suggest that both academic and teamwork-related objectives should be supplied to the students. Second, they suggest providing extensive structuring information for academic tasks. Third, they suggest using the technology to structure and enable well-defined patterns of communication. Next, they suggest some training for learners to assist them in working better as a collective. Fifth, they suggest specifying what are considered to be acceptable behaviors within the context of the team. Sixth, they suggest maintaining frequent task-oriented communications to keep all team members up to date on who is doing what and how far each person is along with their assigned responsibilities. Finally they suggest providing learners with a clear
mission and specify the purpose and importance of working with their teammates to accomplish the team’s goals.

According to Kreijns et al. (2003), lack of social interaction among participants is the primary reason for failure in collaborative learning. Most environments focus on the cognitive and task-related aspects of communication among team members; for effective team processes including trust, however social communications are critical. Leidner and Jarvenpaa (1995) theorize the following roles for information technology in the learning environment: redraw the physical boundaries of the classroom, enable more teamwork, allow learning to occur continuously in a time-independent manner, and enable multi-level, multi-speed knowledge creation.

The prior section examined education in the virtual environment. The learning theory most appropriate to the virtual environment is that of Collaborative Learning. Since Collaborative Learning is itself a social process, leadership, another social process, should have powerful effects on learning in the virtual environment.

**Leadership**

Leadership is an important factor in controlling team processes and inducing favorable team outcomes. Organizational leadership refers to the social influence behaviors/actions that help in coordinating action for the long-term survival of the organization. A leadership system encompasses both individual as well as collective leadership behavior within and across levels of the organization (Avolio, 1999). A
leadership system can be viewed as a potential source of structures that guides a team’s actions, including how the team appropriates technology (Kahai, Sosik & Avolio, 1997).

E-leadership is defined as a social influence process mediated by information technology to effect a change in attitudes, emotions, thinking, behavior and/or performance at the individual, group, or organizational level (Avolio et al 2000). E-leadership can be manifest at any level of the organization and in both one-to-one and one-to-many relationships. E-leadership does not always rest with one person; it can be passed from member to member of the group as needed.

Jarvenpaa and Tanriverdi (2003) indicate that the process of leadership will undergo several changes in the new virtual environment. They state that leadership will be transient, short-scoped, and distributed amongst members of the team. Leadership will move from team member to team member depending on who has the knowledge needed to effectively lead the team in the current task. It is also quite likely in the new network forms, organizational structure will become too complex for one person to be designated as the leader. Some leadership responsibilities will be spread around to all members of the organization. Communication technologies and information systems will become the connections among the members of the network and will influence both for good and for bad, how teams interact and collaborate.

Information technology can affect e-leadership based upon the technology’s structural features and spirit, the group’s internal systems, and the internal culture of the
organization. Information technology has the ability to enable, undermine, or completely disable a leadership system (Avolio et al, 2000).

One of the ways that leadership can assist in the performance of a team is by affecting the team’s appropriations. The appropriations made by the team can influence the structures used by the group including those provided by the leader themselves. Avolio et al (2000) state that virtual teams led by a transformational leader will appropriate technology to enhance collaboration and synergies within teams, leading to higher levels of trust and increasing team performance. Since leadership affects a team’s style of interacting, shared mental models, perceptions of others and the group, and the shared mental models of the group members and all of the above factors also affect appropriation of structures, it is expected that leadership will have a significant impact on the appropriation of structures by the group.

Leadership can also affect trust within the team. Leadership is likely to influence the trust within a team by affecting the perceptions of team member’s ability, benevolence, and integrity (Mayer et al, 1995). Transformational leadership is expected to affect benevolence by helping team members to see the benefits of moving beyond their own needs and assisting the team in the accomplishment of its goals (Avolio et al, 2000). By promoting intellectual stimulation, the transformational leader can encourage a questioning of assumptions and reframing of thinking, which can increase information exchange among team members. This increased information exchange should lead to team members revealing more about their abilities or being more transparent, thus
increasing team trust (Jarvenpaa et al. 1998; Avolio et al. 2000). Transformational leadership can also develop trust through individualized consideration, whereby the leader encourages the evaluation of input provided by every member of the team; and inspirational motivation, whereby the leader expresses confidence in the ability of the team as a whole to accomplish its goals. Transactional leadership can develop trust by building a history of team interactions which helps team members build knowledge about other team member’s ability and integrity.

Another important way that leadership can affect team outcomes is through the establishment of norms of behavior, especially in terms of the use of technology (Malhotra et al. 2007). These norms include determining how often message spaces (e-mail, or bulletin boards) should be checked, keeping the document repository up to date, and proper etiquette for electronic communication. Malhotra et al. (2007) also notes that these norms are not static; they will evolve as the team grows and changes over time.

Kayworth and Leidner (2002) found that the most effective leaders in their studies adaptively appropriated the technology they were provided with, even going so far as to add additional functionalities that helped their team accomplish its goals. Zigurs (2003) states that it is important for leaders to thoroughly articulate both process related and relational roles to the team members. By ensuring that the roles and task requirements of team member are clear to everyone involved, the leader can assist in forming knowledge based trust by increasing the group’s shared understanding of the task (Zaccaro & Bader 2003).
The prior sections have discussed leadership in a virtual environment as provided by the team members themselves. However, research has shown that there are times where the GSS software itself fulfills some of the roles normally performed by a leader on the team.

**Group Support Systems**

There are times when it is possible for a group support system (GSS) to take over some of the roles of a leader. Zigurs (2003) notes that in their work with a GSS that half of their participants identified that the software occupied the role of the Recorder and 40 percent viewed the software as filling the role of Proceduralist. In their case, the GSS kept a record of everything all team communications and provided facilities for process structuring and the keeping of agendas so this is not terribly surprising. Even though the GSS did substitute for some of the roles that the participants expected to take for themselves, they continued to rate the experience positively.

Normal project management software includes functions for specifying deliverables, setting timelines, and identification of resources. For a web site to fully support a virtual team’s interactions, however, it needs to do more than this; it also needs to support team process components and encourage interactions among team members at all levels (Zigurs, 2003). For this reason, it was decided that we would develop a Group Support System. This system was developed in C#.NET including AJAX technology. The database was provided by Microsoft SQL Server 2000.
Recommendations for a Group Support System

Based upon the information found during the literature review, it is possible to identify the features and functions that need to be provided by a Group Support System intended for leadership education. The GSS needs to be able to record information about the goals of the team and its overarching mission (Orvis & Lassiter, 2006; Kirkman et al, 2002), this can most simply be accomplished by allowing for the recording and display of the team’s missions statement and charter. Also, a sense of experimentation needs to be fostered among the users of the site (Malhotra et al, 2007). This can be facilitated by making it difficult for irrevocable changes to be made and allow unintended changes to be undone (Norman, 1990). In addition, GSS need to allow for free expression of ideas on the part of team members (Nonaka, 1994). Expression can be facilitated in many ways, including anonymous polling, and anonymous postings to discussion boards.

The functionality required by an ideal GSS can be broken down into four main groups: communication, project management, roles and feedback to users. Following Orvis and Lassiter (2006) an idea GSS should allow for frequent task-oriented communication as well as structuring and enabling a well-defined pattern of communication. Communication functionality within the GSS should be structured to prevent information asymmetry and the resulting lost of trust (Majchrzak et al, 2004). A GSS should also encourage social communication to assist in the building of trust amongst the team members (Kreijns et al, 2003). GSS should also include rich communication media to assist in the development of a social identity and in conflict
resolution (Daft & Lengel, 1986; Kirkman et al, 2002). Lean communication methods should also be built into GSS to develop task based trust (Daft & Lengel, 1986; Avolio et al, 2001). These two communication methods may conflict with each other (i.e. social differences evident in rich communication undermining task based trust built using lean communication methods), so a balance will need to be struck depending on the type of task assigned to the team (Zigurs & Buckland, 1998). Additionally, both synchronous and asynchronous communications technologies should exist in an ideal GSS (Rains & Scott, 2006). In terms of project management, calendaring is one of the most important functions to be provided by a GSS. This is both because it informs the team explicitly of what it is they are doing and when things need to be done and because it forms a sort of contract indicating what has been agreed upon by all the team members in terms of their obligation to the team (Townsend et al, 1998; Rains & Scott 2006).

Another important project management function to be included in an ideal GSS is that of information processing. This is frequently implemented through a file repository, where different versions of files can be tracked and organized. Additionally a place where a team can record both the internal and external resources available for their use enables further information processing (Zigurs & Buckland, 1998; Orvis & Lassiter, 2006).

The final important piece of functionality in a GSS is that of process structuring. Process structuring includes the recording and display of information about the team’s
task and other associated information. This provides a structure to the task which helps the team to know what needs to be done and when (Zigurs & Buckland, 1998).

The ability to clearly assign users to different roles is critical for a GSS (Zigurs, 2003; Zaccaro & Bader, 2003). By assigning users to roles confusion as to which team member is responsible for which task can be avoided, which helps the processes of the team flow more smoothly and assists in the development of task based trust (Jarvenpaa et al, 1998; Jarvenpaa & Leidner, 1999).

Finally, a GSS need to provide adequate feedback to its users. This feedback should not be limited to indicating when something has gone wrong, but also to when a command or operation has succeeded (Zigurs & Buckland, 1998; Brennan, 1998). Error and confirmation messages are the most common ways that this form of feedback is provided by a GSS. Another form of feedback that can be provided to a user is an indication of what actions are available to the user. These affordances (Norman, 1990) are often implemented in the form of a menu structure, or other layout methods that indicate clickable areas of the screen where the user can go to accomplish the desired task. Table 5 summarizes these recommendations.
Table 5 Summary of recommendations for an ideal GSS

<table>
<thead>
<tr>
<th>Aspects of an Ideal GSS</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functions</td>
<td></td>
</tr>
<tr>
<td>Communication</td>
<td></td>
</tr>
<tr>
<td>Asynchronous Communication</td>
<td>Discussion Boards</td>
</tr>
<tr>
<td>Synchronous Communication</td>
<td>Chat Sessions, Conferencing</td>
</tr>
<tr>
<td>Social Communication</td>
<td>Place for users to post information about themselves.</td>
</tr>
<tr>
<td>Project Management</td>
<td></td>
</tr>
<tr>
<td>Calendaring</td>
<td>Goals, Milestones, Tasks, Meetings</td>
</tr>
<tr>
<td>Information Processing</td>
<td>Folders, Files, Resources</td>
</tr>
<tr>
<td>Process Structuring</td>
<td>Projects, Deliverables, Evaluation Criteria</td>
</tr>
<tr>
<td>Feedback</td>
<td>Success and Failure messages, Clearly laid out controls</td>
</tr>
<tr>
<td>Roles</td>
<td>Item owners, Item contributors</td>
</tr>
<tr>
<td>Practices</td>
<td>Mission Statements, Team Charters, Anonymous Feedback</td>
</tr>
</tbody>
</table>
Project Description

The functionality of the site consists of 21 different items that can be connected in pretty much any configuration allowing for a maximum of flexibility in the operation of the website, and the ability to accommodate a wide range of team compositions and tasks. There is no limit to the number of children a parent item can have; a child, however, can only have one parent.

When a new group of users is going to use the site, the first step is for a site administrator to add an Organization entry to the database. The Organization is the highest level item in the website; it allows multiple teams from the same group to share information among each other. It also provides a level of security. Because a user or any items can only belong to one Organization, there is no way for a user to view information belonging to another Organization. Once the Organization is created, the site administrator then adds Users to the Organization. At this stage Users are assigned to one of two roles: Organization Admin or Organization User. The Organization Admin is able to add new teams and modify certain information about the Organization that the Organization Users cannot.

Once the Organization and its Users have been created, the Users are then allowed to begin using the site. Usage of the site involves the Users connecting new items to the items that already have been provided to them, in most cases this will be the Organization. In the case where a template was used to create the site, however, there will already be a number of items for the Users to work with. To connect a new item on the
website the user will: click on the “Connect New Item” link that is in the same place on every page; then select the type of item they wish to add from the possible children of the current item they are viewing; fill in the requested information and then submit the form. By default, the User that creates the new item is placed in the Admin role for that item and all other users of the parent item are placed in the User role for the new item. The users of the site will then go on and add new items, or edit existing items as needed for the duration of their use of the site.
<table>
<thead>
<tr>
<th>Item Type</th>
<th>Allowed Children</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calendar Entry</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Organization</td>
<td>Calendar Entry, Discussion, Evaluation Criteria, Facilitating Factor, Folder,</td>
</tr>
<tr>
<td></td>
<td>Issue, Lesson Learned, Region, Resource, Poll/Survey, Team, User</td>
</tr>
<tr>
<td>Discussion Comment</td>
<td>Evaluation Criteria, Facilitating Factor, Folder, Issue, Lesson Learned, Resource,</td>
</tr>
<tr>
<td></td>
<td>Poll/Survey</td>
</tr>
<tr>
<td>Deliverable</td>
<td>Calendar Entry, Discussion, Evaluation Criteria, Facilitating Factor, Folder,</td>
</tr>
<tr>
<td></td>
<td>Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Department</td>
<td>Calendar Entry, Discussion, Evaluation Criteria, Facilitating Factor, Folder,</td>
</tr>
<tr>
<td></td>
<td>Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Discussion</td>
<td>Calendar Entry, Discussion Comment, Deliverable, Evaluation Criteria, Facilitating</td>
</tr>
<tr>
<td></td>
<td>Factor, Folder, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Evaluation Criteria</td>
<td>Calendar Entry, Deliverable, Discussion, Facilitating Factor, Folder, Issue,</td>
</tr>
<tr>
<td></td>
<td>Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Facilitating Factor</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Folder, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>File</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Folder</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, File, Folder, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Goal</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Issue, Lesson Learned, Milestone, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Item Type</td>
<td>Allowed Children</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Issue</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Lesson Learned</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Issue, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Milestone</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Project</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Goal, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Region</td>
<td>Calendar Entry, Department, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Issue, Lesson Learned, Resource, Poll/Survey</td>
</tr>
<tr>
<td>Resource</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Issue, Lesson Learned, Poll/Survey</td>
</tr>
<tr>
<td>Survey</td>
<td>Calendar Entry, Discussion, Facilitating Factor, Folder, Issue, Resource</td>
</tr>
<tr>
<td>Task</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Goal, Issue, Lesson Learned, Milestone, Resource,</td>
</tr>
<tr>
<td></td>
<td>Poll/Survey, Task</td>
</tr>
<tr>
<td>Team</td>
<td>Calendar Entry, Deliverable, Discussion, Evaluation Criteria,</td>
</tr>
<tr>
<td></td>
<td>Facilitating Factor, Folder, Goal, Issue, Lesson Learned, Project, Resource,</td>
</tr>
<tr>
<td></td>
<td>Poll/Survey</td>
</tr>
<tr>
<td>User</td>
<td>Calendar Entry, Discussion, Facilitating Factor, Folder, Issue, Lesson Learned,</td>
</tr>
<tr>
<td></td>
<td>Resource, Poll/Survey</td>
</tr>
</tbody>
</table>
Site Presentation

There are nine different layouts for the 21 different items in the website.

Layout 1: Organizations, Departments and Regions

- Name of the item
- The description of the item
- The type of item
- Additional information about the item
- Site navigation menu
- Menu of possible actions for the item
- List of users assigned to the item and their roles
- Listing of items that are connected to this item
Layout 2: Users

**Link to User’s Organization**

**Site navigation menu**

**Name of the User**

**Menu of possible actions for the User**

**Information about the User**

**Photo of the User**

**If a User is viewing their own profile, the change password dialog is shown**

**Listing of items that are connected to this item**
Layout 3: Calendar Entries, Deliverables, Discussion Comments, Evaluation Criteria, Facilitating Factors, Goals, Issues, Lessons Learned, Milestones, Projects, Tasks
Layout 4: Discussions

- Link to the Discussion’s parent
- Topic of the Discussion
- The Discussion’s Comments
- Summary information about the Discussion
- Site navigation menu
- Menu of possible actions for the Discussion
- List of users assigned to the Discussion and their roles
- Listing of items that are connected to this Discussion
Layout 5: Teams

- Link to the Team’s Organization
- Name of the Team
- The Team’s Description, Charter and Compact
- Summary information about the Team

Site navigation menu
Menu of possible actions for the Team
List of users assigned to the Team and their roles
Listing of items that are connected to this Team
Layout 6: Resources

- Link to the Resource’s parent
- Name of the Resource
- The Resource’s Description and associated information
- Summary information about the Resource
- Site navigation menu
- List of users assigned to the Resource and their roles
- Listing of items that are connected to this Resource
- Menu of possible actions for the Resource
Layout 7: Polls

- Link to the Poll’s parent
- Site navigation menu
- The Poll’s question
- Menu of possible actions for the Poll
- Poll options or results
- List of users assigned to the Poll and their roles
- Summary information about the Poll
- Listing of items that are connected to this Poll
Layout 8: Files

- Link to the File’s Folder
- Name of the File
- The File’s description, notes and previous versions
- Summary information about the File
- Site navigation menu
- Menu of possible actions for the File
- List of users assigned to the File and their roles
- Listing of items that are connected to this File
## Layout 9: Folders

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link to the Folder’s parent</td>
<td>Provide a link to the parent folder for easy navigation.</td>
</tr>
<tr>
<td>Name of the Folder</td>
<td>Display the folder name prominently.</td>
</tr>
<tr>
<td>The Folder’s description and contents</td>
<td>Include a brief description of the folder and its contents.</td>
</tr>
<tr>
<td>Site navigation menu</td>
<td>Offer navigation options for different sections of the website.</td>
</tr>
<tr>
<td>Menu of possible actions for the Folder</td>
<td>Present actions that can be performed on the folder, such as editing.</td>
</tr>
<tr>
<td>List of users assigned to the Folder</td>
<td>List all users associated with the folder and their roles.</td>
</tr>
<tr>
<td>Listing of items that are connected</td>
<td>Display items connected to the folder, such as documents or sub-folders.</td>
</tr>
</tbody>
</table>

*Note: The diagram illustrates the layout and features of a folder view in a virtual team website.*
Site Functions

Calendar Entries

Calendar Entries allow for the creation of meetings and other events for the team or organization. The actual Calendar Entry itself only stores information about the location and start and end times of the event or meeting; additional information such as agendas or files to be reviewed is stored in other connected items.

Organizations

This is the highest level item in the site structure. It serves as a way of collecting information about a group of teams, and for the sharing of information among different related teams.

Discussion Comments

Discussion Comments are treated the same as any other item on the website in that they can have other items connected to them. This is functionality that may never be used. Early in the design process it, however, it was decided to not limit the functionality presented to the users, but instead to provide them with as flexible of a site as possible and see what they do with it.

Deliverable

This item stores information about Deliverables for a wide variety of items in the website. Calendar Entries, Discussions, Projects and other can all be configured with a Deliverable to help make clear what outcome is expected and to help the team stay on task working towards the same thing.
Departments

The Departments are for information sharing among users of the site that are in the same Department within their Organization. Departments may not be used in all situations, but exist to allow for the organization of users for groups with large numbers using the site.

Discussions

Discussions are able to be connected to just about every item in the site. This helps to ensure that the users of the site can always communicate with each other and talk about the processes of the team and what the team is trying to accomplish.

Evaluation Criteria

Evaluation Criteria are present on the site to more explicitly define what the team is trying to accomplish, and to make sure that everyone is aware of how what they are trying to accomplish will be judged. This will ensure that everyone on the team is on the same page, and hopefully make the operation of the team more efficient.

Facilitating Factors

Facilitating Factors are included to allow the users of the site to record something that made their work easier.

Files

The website supports versioning of files, allowing multiple versions of the same file to be stored on the site. The files are stored in the database rather than on the file system on the web server. This allows for significantly greater flexibility and capability
in the handling of files since it avoids potential naming conflicts as well as increases security since the files can only be accessed in a tightly controlled manner.

Folders

Folders are able to be connected to nearly every item in the website. This allows for things like meeting agendas and minutes or a piece of work product to be stored right with the associated item, making the site configuration simpler for the users.

Goals

Goals allow the team to specify things that they hope to get accomplished. These may not necessarily be work related, given that it is hoped the users of the site will be developing their leadership, team process goals, or personal development goals will also be recorded in this way.

Issues

Issues are included to allow the users of the site to record something that made their work more difficult, or an obstacle that the team encountered.

Lessons Learned

Lessons Learned allows the users of the site to record insights that they have had during the operation of the team. These would then be shared among the team or other user’s of the site to help the leadership development of the team as a whole. Eventually, these will be collected and compiled into a sort of leadership knowledge base that will be made available to the users of the site.
Milestones

Milestones are used for the team to measure their progress towards a goal. This can either be a work related goal or a development goal.

Projects

In terms of the Team’s employer related work, most of the site activity will be focused at the project and its connected items.

Regions

Regions, like Departments, are used to organize the users of the site and facilitate communication among users who work in the same geographic area.

Resources

Currently there are two different types of Resources configured on the site: Contacts and Web Links. The database, however, is configured to allow users to create their own types of resources.

Poll

Polls are used on the site to allow input from the users in the team’s processes. Polls can be anonymous or not. The timing of display of results can also be controlled. This can be used, for example, to make sure that all users have signed off on a document or task as being completed, or to gain feedback about the team’s processes. There will also be a way to integrate the virtual team site with a full web survey generator allowing for more involved surveys or research to be conducted.
Task

A Task is used to coordinate the efforts of the team. By assigning a Task Owner, ownership of the team’s objectives can be demonstrated facilitating their completion.

Team

Teams are the main organization of users on the website. In most circumstances, the team will be the most important item on the site and most communication between users will occur at this level.

User

This is used to store information about the users of the website. The main piece of functionality offered at this level is the time zone setting, which allows all times on the site to be displayed in the User’s local time zone including updates for the varying Daylight Savings Time rules around the world. A User can also have other items connected to it, allowing for a User to have its own discussion board or folders for example.

Site Structure

It was decided early on to make the structure of the site as flexible as possible in order to keep the structure of the site from influencing the group’s processes. This minimal structure allows an examination of how teams appropriate the different functions available to them as their processes evolve, and also allows for a more developed structure to be provided to team members at the beginning followed by an examination of how they work in the more structured environment. As shown in Figure 1, the only items
whose connections within the site are enforced are those of the Organization, Region, Department, User, Team and Project. The connections of the remaining items are determined by function, Discussions and Folders can be connected nearly anywhere, while Tasks can only be connected to Projects or other Tasks. By minimizing the number of enforced connections within the site the intention was to transfer as many of the structure appropriation decisions to the team members themselves, rather than making those decisions for them which would risk having the site’s structure affect the team’s functions (Rutkowski et al, 2002; Zigurs, 2003).

**Figure 1 Enforced Site Structure**

![Diagram of enforced site structure](image)

**Discussion**

To develop this website, additional communication functionality was built into the standard project management functions. By increasing the number of avenues available for communication within the team, effective team processes and leadership
could be further encouraged. Table 7 takes the three group support software dimensions proposed by Zigurs and Buckland (1988) and places the functionality provided by the software developed for this thesis into the appropriate category. By not emphasizing one category over another, the site should be able to facilitate the accomplishment of any of the types of tasks listed in Table 3.

Table 7 Fit between thesis software and Group Support System dimensions

<table>
<thead>
<tr>
<th>Communication</th>
<th>Process Structuring</th>
<th>Information Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discussion Board</td>
<td>Team</td>
<td>Poll</td>
</tr>
<tr>
<td>Linkage to external communication functionality (Skype, Macromedia Breeze)</td>
<td>Project</td>
<td>File</td>
</tr>
<tr>
<td>Facilitating Factor</td>
<td>Task</td>
<td>Folder</td>
</tr>
<tr>
<td>Issue</td>
<td>Milestone</td>
<td>Resource</td>
</tr>
<tr>
<td>Lesson Learned</td>
<td>Goal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Deliverable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Evaluation Criteria</td>
<td></td>
</tr>
</tbody>
</table>

User Feedback

The nine layouts presented above reflect the second iteration of the site’s presentation layer. Several changes were made to the presentation based upon user feedback obtained during initial trials of the site. The old presentation of the site can be seen in Figure 2. The first change was the rounding of the corners, this is not a significant change, however it does make the site seem more accessible and less sterile. The second change involved making a clearer separation between the information about
the current item itself and the listing of items that are connected to the current item. As shown in Figure 2, in the old presentation the two would flow together, however in the new layout (Layout 5) the information about the team itself (Charter, Compact) is clearly separated from the connected items through placing the list of connected items in a specific area of the page.

The next change made was increasing the complexity of the input validation on the website. In the old version, input validation consisted only of making sure that data was entered into the required fields. In the new version input is also validated to make sure that the correct type of information is entered. For example, the revised site now checks that data recognizable as a date has been entered into a date field, and if the data is invalid, a friendly error message is provided to the user showing the expected date format.

The revised site also now includes more forcing mechanisms (Norman, 1990). In the old version, the date and time entry fields were combined into one text input and users would be expected to enter text similar to “9/10/07 9am”. In the new version this single text input is replaced by the following: one text input for the date, a button they can click which displays a calendar where they can select the date graphically, a drop down list for the hour, a drop down list for the minutes, and a radio button to choose between AM and PM. By asking the user to enter date and time information in this way the site greatly reduces the chances of the user making an error.
In the initial site it was not possible to delete items. This functionality has been added to the new site, along with an element of forcing which makes the user confirm their delete decision before going on. This is done by displaying a confirmation dialog on the screen, and disabling the rest of the controls on the site thus forcing the user to confirm or deny their action before moving on.

One change requested by a large number of users of the initial version of the site was the inclusion of e-mail for notifications about additions or changes to the site. This is accomplished by providing a radio button on both the item creation and edit pages where the user can choose to send a notification e-mail. This is set to “yes” as default, but can be set to “no” in the case of a change to correct spelling or grammar, where a notification to the team of the change is not warranted.

The next change made involved relocating some of the administrative functionality of the site. In the initial version, all administrative functions (add new item, edit current item, edit page layout and edit item permissions) were accessed through a single link on the page. This link was in the same place on every page, and existed for every type of item. It was hoped that this level of consistency would make the operation of the site intuitive to the users; however, this was not the case. The main problem with this implementation was the clicking of the edit link would take the user to a second page where they would make whatever changes they wanted, and then they would click another link to take them back to the item they were editing. These additional clicks led to annoyance and confusion among the users.
The solution was to include “Edit this Item”, “Connect a New Item”, and “Edit Assigned Users” links onto the page itself. When these links were clicked a panel would appear where the user could perform the requested action, and would disappear after the action was completed all without leaving the page. The edit page layout functionality was implemented by including arrows on the connected item panels that would move the panel up, down, or into the other column when clicked. Some of the changes were minor, and some were quite extensive, but all of them together led to a much more easy and pleasant to use website than existed previously.

**Figure 2 Initial Site Layout**

![Initial Site Layout](image)

**Limitations**

Despite the level of functionality provided by the site, it is not without its limitations. One limitation of the website involves its communication functions. As presently written, there is no synchronous communication functionality directly built into the site. Instead, the website is integrated with Skype to provide synchronous audio and
text communication, this fact significantly reduces the richness of the communications enabled by the site itself. There is also no real-time collaborative functionality where team members can edit a document at the same time. This will complicate brain-storming sessions where the ability to quickly record and share ideas with other team members is vital. In some cases, users are able to use Macromedia Breeze for this functionality, however, no integration is provided so materials created in Breeze must be manually transferred back into the team site.
Appendix

Database Structure

Tables

aspnet_Applications

Description

This table stores information the name of the ASP.NET application environment used by the website.

Fields

- ApplicationName – nvarchar 256 – The name of the ASP.NET application
- LoweredApplicationName – nvarchar 256 – The name of the ASP.NET application in all lower case letters
- ApplicationID – uniqueidentifier – The ID of the ASP.NET application
- Description – nvarchar 256 – The description of the ASP.NET application

aspnet_Membership

Description

This table stores information about the Users of the website required for the ASP.NET membership component to function.

Fields

- Application ID – uniqueidentifier – The ID of the ASP.NET application, a foreign key for the aspnet_Applications Table
- UserID – uniqueidentifier – The ASP.NET Application User ID of the User, a foreign key for the aspnet_Users table
- Password – nvarchar 128 – The User’s password, encrypted, hashed or plain text depending on application settings
• PasswordFormat – int – The format of the User’s password. (0 = PlainText, 1 = Hashed, 2 = Encrypted)

• PasswordSalt – nvarchar 128 – The salt value for the hashing of the User’s password.

• MobilePin – nvarchar 16 – The User’s mobile pin and is currently not used by ASP.NET

• Email – nvarchar 256 – The User’s e-mail address

• LoweredEmail – nvarchar 256 – The User’s e-mail address in all lowercase letters

• PasswordQuestion – nvarchar 256 – The question the User is asked if they forget their password

• PasswordAnswer – nvarchar 128 – The expected answer to the User’s PasswordQuestion

• IsApproved – bit – True/False value indicating is the User’s account has been approved by the Application administrator

• IsLockedOut – bit – True/False value indicating if the User’s account has been locked out of the system, usually for exceeding the maximum number of failed login attempts.

• CreatedDate – datetime – Date and time the User’s account was created;

• LastLoginDate – datetime – The date and time the User last logged in

• LastPasswordChangedDate – datetime – The date and time the User’s password was last changed

• LastLockoutDate – datetime – The last date and time the User was last locked out

• FailedPasswordAttemptCount – int – User’s number of consecutive failed login attempts

• FailedPasswordAttempt-WindowStart – datetime – Date and time of User’s first failed login attempt, if FailedPasswordAttemptCount is not zero

• FailedPasswordAnswer-AttemptCount – int – User’s number of consecutive incorrect answers to their PasswordQuestion

• FailedPasswordAnswer-AttemptWindowStart – datetime – Date and time of User’s first failed answer to their PasswordQuestion, if FailedPasswordAnswer-AttemptCount is not zero

• Comment – ntext – additional information about the User

aspnet_Profile
**Description**

This table stored information about the User’s ASP.NET Profile, in the case of this project, their time zone.

**Fields**

- **UserID** – uniqueidentifier – The ASP.NET Application User ID of the User the profile data is associated with, a foreign key for the aspnet_Users table
- **PropertyNames** – ntext – Names of all property values associated with this profile
- **PropertyValuesString** – ntext – Values of the properties that can be stored as text
- **PropertyValuesBinary** – image – Values of the properties that can be stored as binary data
- **LastUpdatedDate** – datetime – The last date and time the profile information was updated

**aspnet_Roles**

**Description**

This table stores information about the different roles in the ASP.NET membership system. In our case they are: Site Administrator, users in this group can add new Organizations, add new Users and manage existing Users; Organization Administrator and Organization User, users in these groups have the same permission levels, and the roles have been separated to allow for a separation of some of the user creation responsibilities from the Site Administrators down the road.

**Fields**

- **Application ID** – uniqueidentifier – The ID of the ASP.NET application these roles are associated with, a foreign key for the aspnet_Applications table
- **RoleID** – uniqueidentifier – The ID of the role
- **RoleName** – nvarchar 256 – The name of the role
- **LoweredRoleName** – nvarchar 256 – The name of the role in all lower case letters
• Description – nvarchar 256 – The description of the role

**aspnet_Users**

**Description**

This table stores information about the ASP.NET application’s users.

**Fields**

• ApplicationID – uniqueidentifier – The ID of the ASP.NET application the user belongs to, a foreign key for the aspnet_Applications table

• UserID – uniqueidentifier – The ASP.NET Application UserID of the user

• UserName – nvarchar 256 – The user’s login to the website

• LoweredUserName – nvarchar 256 – The user’s login in all lower case letters

• MobileAlias – nvarchar 16 – The user’s mobile alias, currently not used by ASP.NET

• IsAnonymous – bit – Indicates whether or not this user account is for the anonymous user on a website (1 = anonymous user, 0 = not the anonymous user)

• LastActivityDate – datetime – The last date and time the user performed an action on the website

**aspnet_UsersInRoles**

**Description**

This table provides the link between the aspnet_Users table and the aspnet_Roles table, allowing for a many-to-many relationship between the records in the two tables.

**Fields**

• UserID – uniqueidentifier – The ASP.NET Application UserID being linked, a foreign key to the aspnet_Users table

• RoleID – uniqueidentifier – The RoleID being linked, a foreign key to the aspnet_Roles table

**tblCalEntry**
Description

This table stores the information about the Calendar Entry items on the website.

Fields

- EntryID – int – The ID of the Calendar Entry
- EntryName – varchar 50 – The name of the Calendar Entry
- EntryDesc – text – The description of the Calendar Entry
- CreationDate – datetime – The date and time the Calendar Entry was created
- LastUpdated – datetime – The date and time the Calendar Entry as last updated
- LastUpdatedBy – int – The VRT UserID of the user that last updated the Calendar Entry
- EventDate – datetime – The date and time of the Calendar Entry
- EventDuration – varchar 50 – The duration of the Calendar Entry stored as an ASP.NET TimeSpan variable
- EventTypeID – int – The ID showing the type of event, a foreign key for the tblCalEntryTypes table
-CreatedBy – int – The VRT UserID of the user that created the Calendar Entry
- VisibleToUsers – bit – Indicates if the Calendar Entry will be visible to users on the website (0 = not visible, 1 = visible)

tblCalEntryAssignments

Description

This table stores the linkages between Calendar Entries and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the Calendar Entry is connected to is populated with the ID of the parent item, all other assignment fields are NULL.

Fields
• EntryAssignmentID – int – The ID of the Calendar Entry's linkage
• EntryID – int – The ID of the Calendar Entry being linked, a foreign key to the tblCalEntry table
• ClientID – int – The ID of the Client the Calendar Entry is linked to, a foreign key to the tblClients table
• RegionID – int – The ID of the Region the Calendar Entry is linked to, a foreign key to the tblRegions table
• DepartmentID – int – The ID of the Department the Calendar Entry is linked to, a foreign key to the tblDepartments table
• TeamID – int – The ID of the Team the Calendar Entry is linked to, a foreign key to the tblTeams table
• UserID – int – The ID of the User the Calendar Entry is linked to, a foreign key to the tblUsers table
• ProjectID – int – The ID of the Project the Calendar Entry is linked to, a foreign key to the tblProjects table
• TaskID – int – The ID of the Task the Calendar Entry is linked to, a foreign key to the tblTasks table
• GoalID – int – The ID of the Goal the Calendar Entry is linked to, a foreign key to the tblGoals table
• MilestoneID – int – The ID of the Milestone the Calendar Entry is linked to, a foreign key to the tblMilestones table
• IssueID – int – The ID of the Issue the Calendar Entry is linked to, a foreign key to the tblIssues table
• FactorID – int – The ID of the Facilitating Factor the Calendar Entry is linked to, a foreign key to the tblFactors table
• ResourceID – int – The ID of the Resource the Calendar Entry is linked to, a foreign key to the tblResources table
• DeliverableID – int – The ID of the Deliverable the Calendar Entry is linked to, a foreign key to the tblDeliverables table
• CriteriaID – int – The ID of the Evaluation Criteria the Calendar Entry is linked to, a foreign key to the tblEvalCriteria table
• SurveyID – int – The ID of the Survey the Calendar Entry is linked to, a foreign key to the tblSurveys table
• LessonID – int – The ID of the Lesson Learned the Calendar Entry is linked to, a foreign key to the tblLessonsLearned table

• DiscussionID – int – The ID of the Discussion the Calendar Entry is linked to, a foreign key to the tblDiscussions table

• FolderID – int – The ID of the Folder the Calendar Entry is linked to, a foreign key to the tblFolders table

• FileID – int – The ID of the File the Calendar Entry is linked to, a foreign key to the tblFiles table

• ParentEntryID – int – The ID of the Calendar Entry the current Calendar Entry is linked to, a foreign key to the tblCalEntry table, this allows for a Calendar Entry to be assigned to another Calendar entry, for example a planning meeting for an upcoming conference

**tblCalEntryTypes**

*Description*

This table describes the different types of Calendar Entries that exist in the application.

Right now there are only two types: Events and Meetings. However this table allows the flexibility to additional types in the future should they be requested.

*Fields*

• EventTypeID – int – The ID of the CalEntryType

• EventType – varchar 50 – The name of the CalEntryType

**tblClients**

*Description*

This table stores the information about the Organizations on the site. The Organization is the highest level object in the site.

*Fields*

• ClientID – int – The ID of the Organization

• ClientName – varchar 100 – The Name of the Organization
• StyleSheet – varchar 50 – The name of the Organization’s stylesheet, this allows for different Organizations to have different color schemes

• ClientDescription – text – The description of the Organization

• ClientMessage – text – A message for Users of the Organization, like a message of the day or a mission statement

• Country – varchar 50 – The country the Organization is headquartered in

• BusinessLine – varchar 50 – The business line of the Organization

• LastUpdated – datetime – The date and time the Organization was last updated

• LastUpdatedBy – int – The VRT UserID of the user that last updated the Organization

tblComments

Description

This table stores the information about the Comments for a Discussion.

Fields

• CommentID – int – The ID of the Discussion Comment

• DiscussionID – int – The ID of the Discussion the Comment is a part of, foreign key to the tblDiscussions table

• ParentID – int – The ID of the Comment the current Comment is assigned to, this functionality is not currently implemented, foreign key to the tblComments table

• SubDate – datetime – The date and time the Comment was submitted

• CommentData – text – The text of the Comment

• SubmitterID – int – The VRT UserID of the user who submitted the Comment

• LastEdited – datetime – The date and time the Comment was last edited

• LastEditedBy – int – The VRT UserID of the user who last edited the Comment

• VisibleToUsers – bit – Indicates if the Comment is visible to the users of the site (0 = not visible, 1 = visible)
**tblDeliverableAssignments**

**Description**

This table stores the linkages between Deliverables and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the Deliverable is connected to is populated with the ID of the parent item, all other assignment fields are NULL.

**Fields**

- DeliverableAssignmentID – int – The ID of the Deliverable’s linkage
- DeliverableID – int – The ID of the current Deliverable, foreign key to the tblDeliverables table
- TeamID – int – The ID of the Team the Deliverable is linked to, a foreign key to the tblTeams table
- ProjectID – int – The ID of the Project the Deliverable is linked to, a foreign key to the tblProjects table
- MilestoneID – int – The ID of the Milestone the Deliverable is linked to, a foreign key to the tblMilestones table
- TaskID – int – The ID of the Task the Deliverable is linked to, a foreign key to the tblTasks table
- EntryID – int – The ID of the Calendar Entry the Deliverable is linked to, a foreign key to the tblCalEntry table
- DiscussionID – int – The ID of the Discussion the Deliverable is linked to, a foreign key to the tblDiscussions table
- LessonID – int – The ID of the Lesson Learned the Deliverable is linked to, a foreign key to the tblLessonsLearned table
- SurveyID – int – The ID of the Survey the Deliverable is linked to, a foreign key to the tblSurveys table
• CriteriaID – int – The ID of the Evaluation Criteria the Deliverable is linked to, a foreign key to the tblEvalCriteria table

• ResourceID – int – The ID of the Resource the Deliverable is linked to, a foreign key to the tblResources table

• FactorID – int – The ID of the Facilitating Factor the Deliverable is linked to, a foreign key to the tblFactors table

• IssueID – int – The ID of the Issue the Deliverable is linked to, a foreign key to the tblIssues table

• FileID – int – The ID of the File the Deliverable is linked to, a foreign key to the tblFiles table

• FolderID – int – The ID of the Folder the Deliverable is linked to, a foreign key to the tblFolders table

• GoalID – int – The ID of the Goal the Deliverable is linked to, a foreign key to the tblGoals table

tblDeliverables

Description

This table stores information about the Deliverables created on the website

Fields

• DeliverableID – int – The ID of the Deliverable

• DeliverableName – varchar 50 – The name of the Deliverable

• CreationDate – datetime – The date and time the Deliverable was created

• LastUpdated – datetime – The date and time the Deliverable was last updated

• LastUpdatedBy – int – the VRT UserID of the user that last updated the Deliverable, a foreign key to the tblUsers table

• DateDue – datetime – The date and time the Deliverable is due

• Status – int – The ID of the Deliverable’s status, a foreign key to the tblStatuses table
• DeliverableDesc – text – The description of the Deliverable
• CreatedBy – int – The VRT UserID of the user that created the Deliverable, a foreign key to the tblUsers table
• VisibleToUsers – bit – Indicates if the Deliverable is visible to the users of the website (0 = not visible, 1 = visible)

**tblDepartments**

*Description*

This table stores the information about the Departments created on the website.

*Fields*

• DepartmentID – int – The ID of the Department
• RegionID – int – The ID of the region the Department is linked to, a foreign key to the tblRegions table
• DepartmentName – varchar 50 – The name of the Department
• DepartmentDesc – text – The description of the Department
• DepartmentLocation – varchar 100 – The geographical location of the department
• LastUpdated – datetime – The date and time the Department was last updated
• LastUpdatedBy – int – The VRT UserID of the User that last updated the Department, a foreign key to the tblUsers table
• CreationDate – datetime – The date and time the Department was created
• CreatedBy – int – The VRT UserID of the User that created the Department, a foreign key to the tblUsers table
• VisibleToUsers – bit – Indicates if the Department will be visible to users (0 = not visible, 1 = visible)

**tblDiscussionAssignments**

*Description*
This table stores the linkages between Discussions and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the Discussion is connected to is populated with the ID of the parent item, all other assignment fields are NULL.

**Fields**

- **DiscussionAssignmentID** – int – The ID of the linkage between the Discussion and the item it is linked to
- **DiscussionID** – int – The ID of the Discussion being linked, a foreign key to the tblDiscussions table
- **ClientID** – int – The ID of the Client the Discussion is linked to, a foreign key to the tblClients table
- **RegionID** – int – The ID of the Region the Discussion is linked to, a foreign key to the tblRegions table
- **DepartmentID** – int – The ID of the Department the Discussion is linked to, a foreign key to the tblDepartments table
- **TeamID** – int – The ID of the Team the Discussion is linked to, a foreign key to the tblTeams table
- **UserID** – int – The VRT UserID of the User the Discussion is linked to, a foreign key to the tblUsers table
- **ProjectID** – int – The ID of the Project the Discussion is linked to, a foreign key to the tblProjects table
- **TaskID** – int – The ID of the Task the Discussion is linked to, a foreign key to the tblTasks table
- **MilestoneID** – int – The ID of the Milestone the Discussion is linked to, a foreign key to the tblMilestones table
- **GoalID** – int – The ID of the Goal the Discussion is linked to, a foreign key to the tblGoals table
• FileID – int – The ID of the File the Discussion is linked to, a foreign key to the tblFiles table
• IssueID – int – The ID of the Issue the Discussion is linked to, a foreign key to the tblIssues table
• FactorID – int – The ID of the Factor the Discussion is linked to, a foreign key to the tblFactors table
• ResourceID – int – The ID of the Resource the Discussion is linked to, a foreign key to the tblResources table
• DeliverableID – int – The ID of the Deliverable the Discussion is linked to, a foreign key to the tblDeliverables table
• CriteriaID – int – The ID of the Evaluation Criteria the Discussion linked to, a foreign key to the tblEvalCriteria table
• SurveyID – int – The ID of the Survey the Discussion is linked to, a foreign key to the tblSurveys table
• LessonID – int – The ID of the Lesson Learned the Discussion is linked to, a foreign key to the tblLessonsLearned table
• FolderID – int – The ID of the Folder the Discussion is linked to, a foreign key to the tblFolders table
• EntryID – int – The ID of the Calendar Entry the Discussion is linked to, a foreign key to the tblCalEntry table

**tblDiscussions**

**Description**

This table stores the information about the Discussions that have been created on the website.

**Fields**

• DiscussionID – int – The ID of the Discussion
• DiscussionTopic – varchar 50 – The topic or title of the Discussion
- OwnerID – int – The VRT UserID of the user that started the discussion, a foreign key to the tblUsers table
- StartDate – datetime – The date and time the Discussion was started
- Archived – int – Indicates if the Discussion has been archived (0 = not archived, 1 = archived)
- LastUpdated – datetime – The date and time the Discussion was last updated
- LastUpdatedBy – int – The VRT UserID of the user that last updated the Discussion
- VisibleToUsers – bit – Indicates if the Discussion will be visible to users of the site (0 = not visible, 1 = visible)

**tblEvalCriteria**

**Description**

This table stores the information associated with the Evaluation Criteria that are created on the website.

**Fields**

- CriteriaID – int – The ID of the Evaluation Criteria
- CreationDate – datetime – The date and time the Evaluation Criteria was created
- LastUpdated – datetime – The date and time the Evaluation Criteria was last updated
- LastUpdatedBy – int – The VRT UserID of the user that last updated the Evaluation Criteria
- CriteriaDesc – text – The description of the Evaluation Criteria
- CreatedBy – int – The VRT UserID of the user that created the Evaluation Criteria
- VisibleToUsers – bit – Indicates if the Evaluation Criteria will be visible to users on the site (0 = not visible, 1 = visible)

**tblEvalCriteriaAssignments**
**Description**

This table stores the linkages between Evaluation Criteria and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the Evaluation Criteria is connected to is populated with the ID of the parent item, all other assignment fields are NULL.

**Fields**

- **CriteriaAssignmentID** – int – The ID of the linkage between the Evaluation Criteria and the item it is linked to
- **CriteriaID** – int – The ID of the Evaluation Criteria being linked, a foreign key to the tblEvaluationCriteria table
- **RegionID** – int – The ID of the Region the Evaluation Criteria is linked to, a foreign key to the tblRegions table
- **DepartmentID** – int – The ID of the Department the Evaluation Criteria is linked to, a foreign key to the tblDepartments table
- **TeamID** – int – The ID of the Team the Evaluation Criteria is linked to, a foreign key to the tblTeams table
- **ProjectID** – int – The ID of the Project the Evaluation Criteria is linked to, a foreign key to the tblProjects table
- **TaskID** – int – The ID of the Task the Evaluation Criteria is linked to, a foreign key to the tblTasks table
- **GoalID** – int – The ID of the Goal the Evaluation Criteria is linked to, a foreign key to the tblGoals table
- **MilestoneID** – int – The ID of the Milestone the Evaluation Criteria is linked to, a foreign key to the tblMilestones table
- **FolderID** – int – The ID of the Folder the Evaluation Criteria is linked to, a foreign key to the tblFolders table
• FileID – int – The ID of the File the Evaluation Criteria is linked to, a foreign key to the tblFiles table

• IssueID – int – The ID of the Issue the Evaluation Criteria is linked to, a foreign key to the tblIssues table

• FactorID – int – The ID of the Factor the Evaluation Criteria is linked to, a foreign key to the tblFactors table

• ResourceID – int – The ID of the Resource the Evaluation Criteria is linked to, a foreign key to the tblResources table

• DeliverableID – int – The ID of the Deliverable the Evaluation Criteria is linked to, a foreign key to the tblDeliverables table

• SurveyID – int – The ID of the Survey the Evaluation Criteria is linked to, a foreign key to the tblSurveys table

• LessonID – int – The ID of the Lesson Learned the Evaluation Criteria is linked to, a foreign key to the tblLessonsLearned table

• DiscussionID – int – The ID of the Discussion the Evaluation Criteria is linked to, a foreign key to the tblDiscussions table

• EntryID – int – The ID of the Calendar Entry the Evaluation Criteria is linked to, a foreign key to the tblCalEntry table

• CommentID – int – The ID of the Discussion Comment the Evaluation Criteria is linked to, a foreign key to the tblComments table

**tblFactors**

**Description**

This table stores information about the Facilitating Factors that have been created on the website. Facilitating Factors are set up to only have a one-to-one linkage instead of a many-to-many linkage allowed for with some of the other items, because of that, the linkage information is stored in the table rather than in a separate table.

**Fields**
- FactorID – int – The ID of the Facilitating Factor
- ClientID – int – The ID of the Client the Facilitating Factor is linked to, a foreign key to the tblClients table
- RegionID – int – The ID of the Region the Facilitating Factor is linked to, a foreign key to the tblRegions table
- DepartmentID – int – The ID of the Department the Facilitating Factor is linked to, a foreign key to the tblDepartments table
- TeamID – int – The ID of the Team the Facilitating Factor is linked to, a foreign key to the tblTeams table
- UserID – int – The VRT UserID of the User the Facilitating Factor is linked to, a foreign key to the tblUsers table
- ProjectID – int – The ID of the Project the Facilitating Factor is linked to, a foreign key to the tblProjects table
- TaskID – int – The ID of the Task the Facilitating Factor is linked to, a foreign key to the tblTasks table
- MilestoneID – int – The ID of the Milestone the Facilitating Factor is linked to, a foreign key to the tblMilestones table
- GoalID – int – The ID of the Goal the Facilitating Factor is linked to, a foreign key to the tblGoals table
- FileID – int – The ID of the File the Facilitating Factor is linked to, a foreign key to the tblFiles table
- IssueID – int – The ID of the Issue the Facilitating Factor is linked to, a foreign key to the tblIssues table
- ResourceID – int – The ID of the Resource the Facilitating Factor is linked to, a foreign key to the tblResources table
- DeliverableID – int – The ID of the Deliverable the Facilitating Factor is linked to, a foreign key to the tblDeliverables table
- CriteriaID – int – The ID of the Evaluation Criteria the Facilitating Factor is linked to, a foreign key to the tblEvalCriteria table
• SurveyID – int – The ID of the Survey the Facilitating Factor is linked to, a foreign key to the tblSurveys table

• LessonID – int – The ID of the Lessons Learned the Facilitating Factor is linked to, a foreign key to the tblLessonsLearned table

• CommentID – int – The ID of the Discussion Comment the Facilitating Factor is linked to, a foreign key to the tblComments table

• DiscussionID – int – The ID of the Discussion the Facilitating Factor is linked to, a foreign key to the tblDiscussions table

• FactorName – varchar 50 – The name of the Facilitating Factor

• FolderID – int – The ID of the Folder the Facilitating Factor is linked to, a foreign key to the tblFolders table

• EntryID – int – The ID of the Calendar Entry the Facilitating Factor is linked to, a foreign key to the tblCalEntry table

• LastUpdated – datetime – The date and time the Evaluation Criteria was last updated

• LastUpdatedBy – int – The VRT UserID of the user that last updated the Evaluation Criteria, a foreign key to the tblUsers table

• CreationDate – datetime – The date and time the Evaluation Criteria was created

• FactorText – text – The text or description of the Facilitating Factor

• CreatedBy – int – The VRT UserID of the user that created the Evaluation Criteria, a foreign key to the tblUsers table

• VisibleToUsers – bit – Indicates if the Facilitating Factor will be visible to the users of the site (0 = not visible, 1 = visible)

**tblFileAssignments**

**Description**

This table allows Files to be linked to Folders. The table is set up to allow for a many-to-many relationship; however this feature is not implemented at this time.

**Fields**
- FileID – int – The ID of the File being linked, a foreign key to the tblFiles table
- FolderID – int – The ID of the Folder the File is linked to, a foreign key to the tblFolders table

**tblFileData**

**Description**

This table stores the actual contents of the Files uploaded to the system. This table allows for different versions of the same file to be stored in the system.

**Fields**

- FileDataID – int – The ID of this version of the File contents
- FileID – int – The ID of the File, a foreign key to the tblFiles table
- UploadedDate – datetime – The date and time this version of the File contents were uploaded
- UploadedBy – int – The VRT UserID of the user that uploaded this version of the File contents, a foreign key to the tblUsers table
- FileData – image – The actual contents of the File
- VersionNotes – text – Notes for this version of the File contents
- LastUpdated – datetime – The date and time this version of the File contents was last updated
- LastUpdatedBy – int – The VRT UserID of the user that last updated this version of the File contents, a foreign key to the tblUsers table
- VisibleToUsers – bit – Indicates if this version of the File contents is visible to users of the site (0 = not visible, 1 = visible)
- IsCurrentVersion – bit – Indicates if this version of the File contents is the current version (0 = not current version, 1 = current version)
- FileLength – bigint – This is the size of the current version of the File contents, in bytes, required for retrieving the file from the database and downloading to the user
• FileContentType – varchar 100 – The MIME type of the current version of the File contents

tblFiles

Description
This table contains the information about each File that is independent of the specific version of the file

Fields
• FileID – int – The ID of the File
• FileName – varchar 200 – The name of the File
• FileDesc – text – The description of the File
• CreationDate – datetime – The date and time the File was created
• CreatedBy – int – The VRT UserID of the user that created the File, a foreign key to the tblUsers table
• LastUpdated – datetime – The date and time the File was last updated
• LastUpdatedBy – int – The VRT UserID of the user that last updated the File, a foreign key to the tblUsers table
• VisibleToUsers – bit – Indicates if the File will be visible to the users of the website (0 = not visible, 1 = visible)

tblFolderAssignments

Description
This table stores the linkages between Folders and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the
Folders is connected to is populated with the ID of the parent item, all other assignment fields are NULL.

**Fields**

- **FolderAssignmentID** – int – The ID of the linkage between the Folder and the Item it is assigned to
- **FolderID** – int – The ID of the Folder being linked, a foreign key to the tblFolders table
- **DeliverableID** – int – The ID of the Deliverable the Folder is linked to, a foreign key to the tblDeliverables table
- **ProjectID** – int – The ID of the Project the Folder is linked to, a foreign key to the tblProjects table
- **ResourceID** – int – The ID of the Resource the Folder is linked to, a foreign key to the tblResources table
- **TeamID** – int – The ID of the Team the Folder is linked to, a foreign key to the tblTeams table
- **TaskID** – int – The ID of the Task the Folder is linked to, a foreign key to the tblTasks table
- **MilestoneID** – int – The ID of the Milestone the Folder is linked to, a foreign key to the tblMilestones table
- **GoalID** – int – The ID of the Goal the Folder is linked to, a foreign key to the tblGoals table
- **DepartmentID** – int – The ID of the Department the Folder is linked to, a foreign key to the tblDepartments table
- **RegionID** – int – The ID of the Region the Folder is linked to, a foreign key to the tblRegions table
- **ClientID** – int – The ID of the Client the Folder is linked to, a foreign key to the tblClients table
- UserID – int – The VRT UserID of the user the Folder is linked to, a foreign key to the tblUsers table

- IssueID – int – The ID of the Issue the Folder is linked to, a foreign key to the tblIssues table

- FactorID – int – The ID of the Factor the Folder is linked to, a foreign key to the tblFactors table

- CriteriaID – int – The ID of the Evaluation Criteria the Folder is linked to, a foreign key to the tblEvalCriteria table

- LessonID – int – The ID of the Lesson Learned the Folder is linked to, a foreign key to the tblLessons Learned table

- CommentID – int – The ID of the Discussion Comment the Folder is linked to, a foreign key to the tblComments table

- DiscussionID – int – The ID of the Discussion the Folder is linked to, a foreign key to the tblDiscussions table

- ParentFolder – int – The ID of the Folder this Folder is linked to, a foreign key to the tblFolders table. Allows for the existence of subfolders.

- SurveyID – int – The ID of the Survey the Folder is linked to, a foreign key to the tblSurveys table

- EntryID – int – The ID of the Calendar Entry the Folder is linked to, a foreign key to the tblCalEntry table

- FileID – int – The ID of the File the Folder is linked to, a foreign key to the tblFiles table

**tblFolders**

**Description**

This table stores the information about the Folders on the website.

**Fields**

- FolderID – int – The ID of the Folder

- FolderName – varchar 50 – The name of the Folder

- FolderDesc – text – The description of the Folder
• CreationDate – datetime – The date and time the Folder was created
• LastUpdated – datetime – The date and time the Folder was last updated
• LastUpdatedBy – int – The VRT UserID of the user that last updated the folder, a foreign key to the tblUsers table
• CreatedBy – int – The VRT UserID of the user that created the folder, a foreign key to the tblUsers table
• VisibleToUsers – bit – Indicates if the Folder is visible to the users of the site (0 = not visible, 1 = visible)

tblFunctions

Description

This table lists the options for the functions users can perform on the system, like view item, add item or edit item. This is an aspect of the VRT permissions system.

Fields

• FunctionID – int – The ID of the Function
• FunctionName – varchar 50 – The name of the Function

tblFunctionToType

Description

This table links the different functions to their corresponding item types, i.e. View Folder to Folder, etc.

Fields

• FunctionID – int – The ID of the Function, a foreign key to the tblFunctions table
• TypeID – int – The ID of the item type, a foreign key to the tblTypes table

tblGoals
**Description**

The table contains information about the Goals contained on the system. Since goals are very specific, this assignment information is contained in this table allowing for only one-to-one relationships.

**Fields**

- GoalID – int – The ID of the Goal
- TaskID – int – The ID of the Task the Goal is linked to, a foreign key to the tblTasks table
- ProjectID – int – The ID of the Project the Goal is linked to, a foreign key to the tblProjects table
- TeamID – int – The ID of the Team the Goal is linked to, a foreign key to the tblTeams table
- GoalName – varchar 50 – The name of the Goal
- EstComplete – datetime – The date and time the Goal is estimated to be completed
- StartDate – datetime – The date and time work on the Goal will begin
- Status – int – The ID of the Goal’s status, a foreign key to the tblStatuses table
- GoalDesc – text – The description of the Goal
- CreationDate – datetime – The date and time the Goal was created
- LastUpdated – datetime – The date and time the Goal was last updated
- LastUpdatedBy – int – The VRT UserID of the user that last updated the Goal, a foreign key to the tblUsers table
- ActualComplete – datetime – The date and time the Goal was actually completed
- CreatedBy – int – The VRT UserID of the user that created the Goal
- VisibleToUsers – bit – Indicates if the Goal will be visible to users on the site (0 = not visible, 1 = visible)
tblIssues

Description

This table stores information about the Issues that have been created on the website. Facilitating Issues are set up to only have a one-to-one linkage instead of a many-to-many linkage allowed for with some of the other items, because of that, the linkage information is stored in the table rather than in a separate table.

Fields

- IssueID – int – The ID of the Issue
- ClientID – int – The ID of the Client the Issue is linked to, a foreign key to the tblClients table
- RegionID – int – The ID of the Region the Issue is linked to, a foreign key to the tblRegions table
- DepartmentID – int – The ID of the Department the Issue is linked to, a foreign key to the tblDepartments table
- TeamID – int – The ID of the Team the Issue is linked to, a foreign key to the tblTeams table
- UserID – int – The VRT UserID of the User the Issue is linked to, a foreign key to the tblUsers table
- ProjectID – int – The ID of the Project the Issue is linked to, a foreign key to the tblProjects table
- TaskID – int – The ID of the Task the Issue is linked to, a foreign key to the tblTasks table
- MilestoneID – int – The ID of the Milestone the Issue is linked to, a foreign key to the tblMilestones table
- GoalID – int – The ID of the Goal the Issue is linked to, a foreign key to the tblGoals table
- FileID – int – The ID of the File the Issue is linked to, a foreign key to the tblFiles table
• FactorID – int – The ID of the Factor the Issue is linked to, a foreign key to the tblFactors table
• ResourceID – int – The ID of the Resource the Issue is linked to, a foreign key to the tblResources table
• DeliverableID – int – The ID of the Deliverable the Issue is linked to, a foreign key to the tblDeliverables table
• CriteriaID – int – The ID of the Evaluation Criteria the Issue is linked to, a foreign key to the tblEvalCriteria table
• SurveyID – int – The ID of the Survey the Issue is linked to, a foreign key to the tblSurveys table
• LessonID – int – The ID of the Lesson Learned the Issue is linked to, a foreign key to the tblLessonsLearned table
• CommentID – int – The ID of the Discussion Comment the Issue is linked to, a foreign key to the tblComments table
• DiscussionID – int – The ID of the Discussion the Issue is linked to, a foreign key to the tblDiscussions table
• IssueText – text – The text or description of the Issue
• FolderID – int – The ID of the Folder the Issue is linked to, a foreign key to the tblFolders table
• EntryID – int – The ID of the Calendar Entry the Issue is linked to, a foreign key to the tblCalEntry table
• LastUpdated – datetime – The date and time the Issue was last updated
• LastUpdatedBy – int – The VRT UserID of the User that last updated the Issue
• CreationDate – datetime – The date and time the Issue was created
• IssueName – varchar 50 – The name of the Issue
• CreatedBy – int – The VRT UserID of the User that created the Issue
• VisibleToUsers – bit – Indicates if the Issue is visible to users on the website (0 = not visible, 1 = visible)
**tblLessonLearnedAssignments**

*Description*

This table stores the linkages between Lessons Learned and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the Lesson Learned is connected to is populated with the ID of the parent item, all other assignment fields are NULL.

*Fields*

- **LessonAssignmentID** – int – The ID of the linkage between the Lesson Learned and the item it is assigned to
- **LessonID** – int – The ID of the Lesson Learned being linked, a foreign key to the tblLessonsLearned table
- **ClientID** – int – The ID of the Client the Lesson Learned is linked to, a foreign key to the tblClients table
- **RegionID** – int – The ID of the Region the Lesson Learned is linked to, a foreign key to the tblRegions table
- **DepartmentID** – int – The ID of the Department the Lesson Learned is linked to, a foreign key to the tblDepartments table
- **TeamID** – int – The ID of the Team the Lesson Learned is linked to, a foreign key to the tblTeams table
- **UserID** – int – The VRT UserID of the user the Lesson Learned is linked to, a foreign key to the tblUsers table
- **ProjectID** – int – The ID of the Project the Lesson Learned is linked to, a foreign key to the tblProjects table
- **TaskID** – int – The ID of the Task the Lesson Learned is linked to, a foreign key to the tblTasks table
• GoalID – int – The ID of the Goal the Lesson Learned is linked to, a foreign key to the tblGoals table

• MilestoneID – int – The ID of the Milestone the Lesson Learned is linked to, a foreign key to the tblMilestones table

• FolderID – int – The ID of the Folder the Lesson Learned is linked to, a foreign key to the tblFolders table

• FileID – int – The ID of the File the Lesson Learned is linked to, a foreign key to the tblFiles table

• IssueID – int – The ID of the Issue the Lesson Learned is linked to, a foreign key to the tblIssues table

• FactorID – int – The ID of the Facilitating Factor the Lesson Learned is linked to, a foreign key to the tblFactors table

• ResourceID – int – The ID of the Resource the Lesson Learned is linked to, a foreign key to the tblResources table

• DeliverableID – int – The ID of the Deliverable the Lesson Learned is linked to, a foreign key to the tblDeliverables table

• CriteriaID – int – The ID of the Evaluation Criteria the Lesson Learned is linked to, a foreign key to the tblEvalCriteria table

• SurveyID – int – The ID of the Survey the Lesson Learned is linked to, a foreign key to the tblSurveys table

• DiscussionID – int – The ID of the Discussion the Lesson Learned is linked to, a foreign key to the tblDiscussions table

• CommentID – int – The ID of the Discussion Comment the Lesson Learned is linked to, a foreign key to the tblComments table

• EntryID – int – The ID of the Calendar Entry the Lesson Learned is linked to, a foreign key to the tblCalEntry table

tblLessonsLearned

Description

This table stores information about the Lessons Learned created on the website
**Fields**

- **LessonID** – int – The ID of the Lesson Learned
- **LessonText** – text – The text or description of the Lesson Learned
- **CreationDate** – datetime – The date and time the Lesson Learned was created
- **LastUpdated** – datetime – The date and time the Lesson Learned was last updated
- **LastUpdatedBy** – int – The VRT UserID of the user that last updated the Lesson Learned
- **LessonName** – varchar 50 – The name of the Lesson Learned
- **CreatedBy** – int – The VRT UserID of the user that created the Lesson Learned
- **VisibleToUsers** – bit – Indicates if the Lesson Learned will be visible to users on the website (0 = not visible, 1 = visible)

**tblMilestones**

**Description**

This table contains information about the Milestones created on the system. Milestones are specific enough to not need many-to-many relationships, so the assignment information is contained in this table.

**Fields**

- **MilestoneID** – int – The ID of the Milestone
- **TaskID** – int – The ID of the Task the Milestone is linked to, a foreign key to the tblTasks table
- **GoalID** – int – The ID of the Goal the Milestone is linked to, a foreign key to the tblGoals table
- **Milestone** – varchar 100 – The name of the milestone
- **EstComplete** – datetime – The date and time the Milestone is estimated to be completed
- **MilestoneDesc** – text – The description of the milestone
• Status – int – The ID of the Milestone’s status, a foreign key to the tblStatuses table
• CreationDate – datetime – The date and time the Milestone was created
• LastUpdated – datetime – The date and time the Milestone was last updated
• LastUpdatedBy – int – The VRT UserID of the user that last updated the Milestone
• ActualComplete – datetime – The date and time the Milestone was actually completed
•CreatedBy – int – The VRT UserID of the user that created the Milestone
• VisibleToUsers – bit – Indicates if the Milestone will be visible to users of the site (0 = not visible, 1 = visible)

**tblObjectRoleAssignments**

*Description*

This table stores the linkages between Roles in the database (Organization Admin, Organization User) and the items in the database (Organizations, Folder). By specifying these linkages we can create specialized roles for some items that aren’t available to other items, i.e. one group wants to have moderators in their discussions, but another group doesn’t, we can assign the Discussion Moderator role to the first group’s discussions only.

*Fields*

• RoleAssignmentID – int – the ID of the Role to Item linkage
• ObjectID – int – The ID of the item being linked, this serves as a foreign key to whichever table corresponds to the item type, this is determined using the tblRoleToType table.
• RoleID – int – The ID of the Role being linked, a foreign key to the tblRoles table

**tblObjectTypeToPageLayoutType**

*Description*
This table links the type of item with its layout of information on the website, i.e. width of the informational tables, if a summary table is displayed, etc.

**Fields**

- **TypeID** – int – The ID of the type of Item being displayed, a foreign key to the tblTypes table
- **PageLayoutTypeID** – int – The ID of the PageLayout used to display the item’s information on the screen, a foreign key to the tblPageLayoutTypes table

**tblPageLayout**

**Description**

This table controls the layout of the tables in the Connected Items section of the page.

**Fields**

- **PageAssignmentID** – int – The ID of the Connected Item assignment
- **ParentID** – int – The ID of the item being displayed on the current page, a foreign key to the table corresponding to the ParentTypeID value
- **ParentTypeID** – int – The ID of the type of item being displayed on the current page, a foreign key to the tblTypes table
- **SummaryObjectTypeID** – int – The ID of the type of item being displayed in the Connected Items area, a foreign key to the tblTypes table
- **WhichColumn** – int – Indicates which column this item’s table will be displayed in (1 = left column, 2 = right column)
- **DispOrder** – int – The order in which this item’s table will be displayed
- **Hidden** – int – Indicates if this item’s table will be hidden (0 = not hidden, 1 = hidden)

**tblPageLayoutTypes**

**Description**
This table stores descriptions of the different page layout types. It is not really used by the system itself, it is for reference by the programmers so they can identify the values in the tblObjectTypeToPageLayoutType table.

**Fields**

- PageLayoutTypeID – int – The ID of the page layout type
- PageLayoutTypeDesc – varchar 200 – The description of the page layout type

**tblParentToChild**

**Description**

This table specifies which types of items are allowed as the children of a given type of item. This allows us to constrain things like making sure that a Region can only be assigned to a Client, or that a File can only be assigned to a Folder.

**Fields**

- ParentTypeID – int – The ID of the type of item the allowed children are being specified for, a foreign key to the tblTypes table
- ChildTypeID – int – The ID of the type of item being allowed as a child of the parent item, a foreign key to the tblTypes table

**tblPollDisplayModes**

**Description**

This table describes the different Poll Display Modes that are available, namely if the results will be displayed immediately after a user submits a response, or if results will be displayed only after all users have submitted responses.

**Fields**

- DisplayModeID – int – The ID of the Display Mode
• DisplayMode – varchar 2000 – The description of the Display Mode

**tblProjects**

**Description**

This table contains information about the Projects created on the website. Since Projects can only be assigned to team and a many-to-many relationship doesn’t make sense, the assignment information is contained within the table.

**Fields**

• ProjectID – int – The ID of the project
• TeamID – int – The ID of the Team the Project is linked to, a foreign key to the tblTeams table
• ProjectName – varchar 50 – The name of the Project
• ProjectDesc – text – The description of the Project
• EstComplete – datetime – The date and time the Project is estimated to be completed
• StartDate – datetime – The date and time the Project will be begun
• Status – int – The ID of the Project’s status, a foreign key to the tblStatuses table
• CreationDate – datetime – The date and time the project was created
• ActualComplete – datetime – The date and time the project was actually completed
• LastUpdated – datetime – The date and time the project was last updated
• LastUpdatedBy – int – The VRT UserID of the user that last updated the Project
• CreatedBy – int – The VRT UserID of the user that created the Project
• VisibleToUsers – bit – Indicates if the project will be visible to users (0 = not visible, 1 = visible)

**tblRegions**
Description

This table contains information about the Regions created in the website. Since Regions can only be assigned to Clients, and a many-to-many relationship doesn’t make sense, the assignment information is stored in this table.

Fields

- RegionID – int – The ID of the Region
- ClientID – int – The ID of the Client the Region is linked to, a foreign key to the tblClients table
- RegionName – varchar 50 – The name of the Region
- RegionDesc – text – The description of the Region
- RegionLocation – varchar 100 – The geographic location of the Region
- LastUpdated – datetime – The date and time the Region was last updated
- LastUpdatedBy – int – The VRT UserID of the user that last updated the Region
- CreationDate – datetime – The date and time the Region was created
- CreatedBy – int – The VRT UserID of the user that created the Region
- VisibleToUsers – bit – Indicates if the Region will be visible to users on the site (0 = not visible, 1 = visible)

Description

This table stores the linkages between Resources and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the Resource is connected to is populated with the ID of the parent item, all other assignment fields are NULL.
**Fields**

- **ResourceAssignmentID** – int – The ID of the linkage between the Resource and the item it is connected to
- **ResourceId** – int – The ID of the Resource being linked, a foreign key to the tblResources table
- **TeamID** – int – The ID of the Team the Resource is linked to, a foreign key to the tblTeams table
- **ProjectID** – int – The ID of the Project the Resource is linked to, a foreign key to the tblProjects table
- **TaskID** – int – The ID of the Task the Resource is linked to, a foreign key to the tblTasks table
- **GoalID** – int – The ID of the Goal the Resource is linked to, a foreign key to the tblGoals table
- **ClientID** – int – The ID of the Client the Resource is linked to, a foreign key to the tblClients table
- **RegionID** – int – The ID of the Region the Resource is linked to, a foreign key to the tblRegions table
- **DepartmentID** – int – The ID of the Department the Resource is linked to, a foreign key to the tblDepartments table
- **UserID** – int – The VRT UserID of the user the Resource is linked to, a foreign key to the tblUsers table
- **DiscussionID** – int – The ID of the Discussion the Resource is linked to, a foreign key to the tblDiscussions table
- **MilestoneID** – int – The ID of the Milestone the Resource is linked to, a foreign key to the tblMilestones table
- **IssueID** – int – The ID of the Issue the Resource is linked to, a foreign key to the tblIssues table
- **FactorID** – int – The ID of the Facilitating Factor the Resource is linked to, a foreign key to the tblFactors table
• DeliverableID – int – The ID of the Deliverable the Resource is linked to, a foreign key to the tblDeliverables table

• CriteriaID – int – The ID of the Evaluation Criteria the Resource is linked to, a foreign key to the tblEvalCriteria table

• SurveyID – int – The ID of the Survey the Resource is linked to, a foreign key to the tblSurveys table

• LessonID – int – The ID of the Lesson Learned the Resource is linked to, a foreign key to the tblLessons table

• EntryID – int – The ID of the Calendar Entry the Resource is linked to, a foreign key to the tblCalEntry table

• CommentID – int – The ID of the Discussion Comment the Resource is linked to, a foreign key to the tblComments table

• FileID – int – The ID of the File the Resource is linked to, a foreign key to the tblFiles table

• FolderID – int – The ID of the Folder the Resource is linked to, a foreign key to the tblFolders table

**tblResourceFieldTypes**

**Description**

This table allows for the different fields of information in a Resource to have different rules governing their formatting when displayed on a web page and their input validation. When we ask the User for a date we can make sure that the text they enter the box is actually a valid date, or when they enter a Time Zone we can display a drop down list with the possible Time Zones displayed.

**Fields**

• ResourceFieldTypeID – int – The ID of the Resource Field Type

• ResourceFieldType – varchar 200 – The Resource Field Type
**tblResources**

**Description**

This table stores information about Resources created on the web site. Since there are multiple types of Resources, this table needed to be able to handle Resources with different numbers of fields and different types of data associated with them. Consequently, this table largely consists of 20 “text” fields whose name and data type information are specified by the tblResourceTypes table.

**Fields**

- ResourceID – int – The ID of the Resource
- ResourceTypeID – int – The ID of this Resource’s Resource Type, a foreign key to the tblResourceTypes table
- ResourceName – varchar 50 – The name of the Resource
- CreationDate – datetime – The date and time the Resource was created
- LastUpdated – datetime – The date and time the Resource was last updated
- LastUpdatedBy – int – The VRT UserID of the user that last updated the Resource
- CreatedBy – int – The VRT UserID of the user that created the Resource
- Field1Data – text – The data in the Resource’s first field
- Field2Data – text – The data in the Resource’s second field
- Field3Data – text – The data in the Resource’s third field
- Field4Data – text – The data in the Resource’s fourth field
- Field5Data – text – The data in the Resource’s fifth field
- Field6Data – text – The data in the Resource’s sixth field
- Field7Data – text – The data in the Resource’s seventh field
- Field8Data – text – The data in the Resource’s eighth field
- Field9Data – text – The data in the Resource’s ninth field
- Field10Data – text – The data in the Resource’s tenth field
- Field11Data – text – The data in the Resource’s eleventh field
- Field12Data – text – The data in the Resource’s twelfth field
- Field13Data – text – The data in the Resource’s thirteenth field
- Field14Data – text – The data in the Resource’s fourteenth field
- Field15Data – text – The data in the Resource’s fifteenth field
- Field16Data – text – The data in the Resource’s sixteenth field
- Field17Data – text – The data in the Resource’s seventeenth field
- Field18Data – text – The data in the Resource’s eighteenth field
- Field19Data – text – The data in the Resource’s nineteenth field
- Field20Data – text – The data in the Resource’s twentieth field
- ResourceDesc – text – The description of the Resource
- VisibleToUsers – bit – Indicates if the Resource will be visible to users on the website (0 = not visible, 1 = visible)

**tblResourceTypes**

**Description**

This table contains information about the different types of Resources that exist on the website. It stores the name of the field, as well as the type of field for each type of Resource.

**Fields**

- ResourceTypeID – int – The ID of the Resource Type
- ResourceType – varchar 50 – The name of the Resource Type
• Field1Name – varchar 50 – The name of the Resource’s first field

• Field1Type – int – The ID of the Resource Field Type of the Resource’s first field, a foreign key to the tblResourceFieldTypes table

• Field2Name – varchar 50 – The name of the Resource’s second field

• Field2Type – int – The ID of the Resource Field Type of the Resource’s second field, a foreign key to the tblResourceFieldTypes table

• Field3Name – varchar 50 – The name of the Resource’s third field

• Field3Type – int – The ID of the Resource Field Type of the Resource’s third field, a foreign key to the tblResourceFieldTypes table

• Field4Name – varchar 50 – The name of the Resource’s fourth field

• Field4Type – int – The ID of the Resource Field Type of the Resource’s fourth field, a foreign key to the tblResourceFieldTypes table

• Field5Name – varchar 50 – The name of the Resource’s fifth field

• Field5Type – int – The ID of the Resource Field Type of the Resource’s fifth field, a foreign key to the tblResourceFieldTypes table

• Field6Name – varchar 50 – The name of the Resource’s sixth field

• Field6Type – int – The ID of the Resource Field Type of the Resource’s sixth field, a foreign key to the tblResourceFieldTypes table

• Field7Name – varchar 50 – The name of the Resource’s seventh field

• Field7Type – int – The ID of the Resource Field Type of the Resource’s seventh field, a foreign key to the tblResourceFieldTypes table

• Field8Name – varchar 50 – The name of the Resource’s eigth field

• Field8Type – int – The ID of the Resource Field Type of the Resource’s eigth field, a foreign key to the tblResourceFieldTypes table

• Field9Name – varchar 50 – The name of the Resource’s ninth field

• Field9Type – int – The ID of the Resource Field Type of the Resource’s ninth field, a foreign key to the tblResourceFieldTypes table
- Field10Name – varchar 50 – The name of the Resource’s tenth field
- Field10Type – int – The ID of the Resource Field Type of the Resource’s tenth field, a foreign key to the tblResourceFieldTypes table
- Field11Name – varchar 50 – The name of the Resource’s eleventh field
- Field11Type – int – The ID of the Resource Field Type of the Resource’s eleventh field, a foreign key to the tblResourceFieldTypes table
- Field12Name – varchar 50 – The name of the Resource’s twelfth field
- Field12Type – int – The ID of the Resource Field Type of the Resource’s twelfth field, a foreign key to the tblResourceFieldTypes table
- Field13Name – varchar 50 – The name of the Resource’s thirteenth field
- Field13Type – int – The ID of the Resource Field Type of the Resource’s thirteenth field, a foreign key to the tblResourceFieldTypes table
- Field14Name – varchar 50 – The name of the Resource’s fourteenth field
- Field14Type – int – The ID of the Resource Field Type of the Resource’s fourteenth field, a foreign key to the tblResourceFieldTypes table
- Field15Name – varchar 50 – The name of the Resource’s fifteenth field
- Field15Type – int – The ID of the Resource Field Type of the Resource’s fifteenth field, a foreign key to the tblResourceFieldTypes table
- Field16Name – varchar 50 – The name of the Resource’s sixteenth field
- Field16Type – int – The ID of the Resource Field Type of the Resource’s sixteenth field, a foreign key to the tblResourceFieldTypes table
- Field17Name – varchar 50 – The name of the Resource’s seventeenth field
- Field17Type – int – The ID of the Resource Field Type of the Resource’s seventeenth field, a foreign key to the tblResourceFieldTypes table
- Field18Name – varchar 50 – The name of the Resource’s eighteenth field
- Field18Type – int – The ID of the Resource Field Type of the Resource’s eighteenth field, a foreign key to the tblResourceFieldTypes table
• Field19Name – varchar 50 – The name of the Resource’s nineteenth field

• Field19Type – int – The ID of the Resource Field Type of the Resource’s nineteenth field, a foreign key to the tblResourceFieldTypes table

• Field20Name – varchar 50 – The name of the Resource’s twentieth field

• Field20Type – int – The ID of the Resource Field Type of the Resource’s twentieth field, a foreign key to the tblResourceFieldTypes table

tblRoles

Description

This table stores the information about each Role for each type of Item on the website. This is a complete listing of all Roles on the site, so not all will necessarily be assigned to all the corresponding items.

Fields

• RoleID – int – The ID of the Role

• RoleName – varchar 50 – The name of the Role

• Standard – bit – Indicates if the Role is automatically assigned to all new items of the type specified in the tblRoleToType table (0 = not assigned, 1 = assigned)

• AssignToCreator – bit – Indicates if the Role is assigned to the creator of the new item (0 = not assigned, 1 = assigned)

• AssignToUser – bit – Indicates if the Role is assigned to the other users assigned to the item, not the creator (0 = not assigned, 1 = assigned)

tblRoleToFunction

Description

This table establishes the linkages between the Roles and the Functions they allow users to perform, i.e. the Admin role for an item would link to both the “edit” and “view” functions,
while the User role for an item would only link to “view”. This table allows for a many-to-many relationship between Roles and Functions

**Fields**

- RoleID – int – The ID of the Role the Function is linked to, a foreign key to the tblRoles table
- FunctionID – int – The ID of the Function being linked, a foreign key to the tblFunctions table

**tblRoleToType**

**Description**

This table establishes the linkages between the Roles and the types of Items they pertain to, i.e. links the “Folder Admin” and “Folder User” roles to the “Folder” Item Type. This table structure allows for many-to-many relationships, however, only many-to-one relationships are implemented (many Roles to 1 Item Type).

**Fields**

- RoleID – int – The ID of the Role being linked, a foreign key to the tblRoles table
- TypeID – int – The type of Item being linked to, a foreign key to the tblTypes table

**tblStatuses**

**Description**

This table contains a list of the possible Statuses for items like Projects, Milestones and Goals.

**Fields**

- StatusID – int – The ID of the Status
- StatusDesc – varchar 50 – The description of the Status
**tblSurveyAssignments**

**Description**

This table stores the linkages between Resources and the other items that they are assigned to within the application, this table’s structure allows for many-to-many linking although that is not currently implemented within the website. The column corresponding to the type of item the Resource is connected to is populated with the ID of the parent item, all other assignment fields are NULL.

**Fields**

- **SurveyAssignmentID** – int – The ID of the linkage between the Survey and the item it is assigned to
- **SurveyID** – int – The ID of the Survey being linked, a foreign key to the tblSurveys table
- **ClientID** – int – The ID of the Client the Survey is linked to, a foreign key to the tblClients table
- **RegionID** – int – The ID of the Region the Survey is linked to, a foreign key to the tblRegions table
- **DepartmentID** – int – The ID of the Department the Survey is linked to, a foreign key to the tblDepartments table
- **TeamID** – int – The ID of the Team the Survey is linked to, a foreign key to the tblTeams table
- **UserID** – int – The VRT UserID of the user the Survey is linked to, a foreign key to the tblUsers table
- **ProjectID** – int – The ID of the Project the Survey is linked to, a foreign key to the tblProjects table
- **TaskID** – int – The ID of the Task the Survey is linked to, a foreign key to the tblTasks table
- **MilestoneID** – int – The ID of the Milestone the Survey is linked to, a foreign key to the tblMilestones table
• **GoalID** – int – The ID of the Goal the Survey is linked to, a foreign key to the tblGoals table

• **FileID** – int – The ID of the File the Survey is linked to, a foreign key to the tblFiles table

• **DiscussionID** – int – The ID of the Discussion the Survey is linked to, a foreign key to the tblDiscussions table

• **IssueID** – int – The ID of the Issue the Survey is linked to, a foreign key to the tblIssues table

• **FactorID** – int – The ID of the Facilitating Factor the Survey is linked to, a foreign key to the tblFactors table

• **ResourceID** – int – The ID of the Resource the Survey is linked to, a foreign key to the tblResources table

• **DeliverableID** – int – The ID of the Deliverable the Survey is linked to, a foreign key to the tblDeliverables table

• **LessonID** – int – The ID of the Lesson Learned the Survey is linked to, a foreign key to the tblLessonsLearned table

• **CriteriaID** – int – The ID of the Evaluation Criteria the Survey is linked to, a foreign key to the tblEvalCriteria table

• **CommentID** – int – The ID of the Discussion Comment the Survey is linked to, a foreign key to the tblComments table

• **FolderID** – int – The ID of the FolderID the Survey is linked to, a foreign key to the tblFolders table

• **EntryID** – int – The ID of the Calendar Entry the Survey is linked to, a foreign key to the tblCalEntry table

**tblSurveychoices**

**Description**

This table stores the possible answers for the Surveys on the website.

**Fields**

• **SurveyChoiceID** – int – The ID of the Survey Choice
• Survey Choice – varchar 2000 – The text of the Survey Choice

• DispOrder – int – The order in which the Survey Choices will be displayed for the Survey specified in the SurveyID field, in ascending order (1 is displayed first, 2 is displayed next…)

• SurveyID – int – The ID of the Survey these Choices will be displayed for, a foreign key to the tblSurveys table

**tblSurveyData**

**Description**

This table stores the User’s answers to the Surveys that have been taken on the website

**Fields**

• DataID – int – The ID of this response to a Survey

• SurveyID – int – The ID of the Survey this is a response for, a foreign key to the tblSurveys table

• UserID – int – The VRT UserID of the User who submitted this response to a Survey, a foreign key to the tblUsers table

• Choice – int – The ID of the Choice selected by the User, a foreign key to the tblSurveyChoices table

**tblSurveys**

**Description**

This table contains information about the Surveys that have been created on the website.

**Fields**

• SurveyID – int – The ID of the Survey

• SurveyQuestion – varchar 2000 – The question asked by the Survey

• Anonymous – bit – Indicates if the Survey is anonymous (0 = not anonymous, 1 = anonymous)
- `CreationDate` – datetime – The date and time the Survey was created
- `CreatedBy` – int – The VRT UserID of the user that created the Survey
- `LastUpdated` – datetime – The date and time the Survey was last updated
- `LastUpdatedBy` – int – The VRT UserID of the user that last updated the Survey
- `ExpirationDate` – datetime – The date and time the Survey stop collecting responses
- `StartDate` – datetime – The date and time the Survey begins collecting responses
- `DisplayMode` – int – Indicates if the Survey will display results after a user has submitted their response, or will wait until all users have submitted responses, a foreign key to the `tblPollDisplayModes` table
- `VisibleToUsers` – bit – Indicates if the Survey is visible to users on the website (0 = not visible, 1 = visible)

**tblTasks**

**Description**

This table contains information about the Tasks that have been created on the website. Tasks are specific enough that allowing for a many-to-many relationship is not required, so the linkage information is only stored in this table, allowing for only one-to-one relationships.

**Fields**

- `TaskID` – int – The ID of the Task
- `ProjectID` – int – The ID of the Project the Task is linked to, a foreign key to the `tblProjects` table
- `ParentTaskID` – int – The ID of the Task this Task is linked to, creating a subtask, a foreign key to the `tblTasks` table
- `TaskName` – varchar 50 – The name of the Task
- `TaskDesc` – text – The description of the Task
- `StartDate` – datetime – The date and time work on the Task will begin
• EstComplete – datetime – The date and time work on the Task is estimated to be complete
• Status – int – The ID of the Task’s status, a foreign key to the tblStatuses table
• CreationDate – datetime – The date and time the Task was created
• LastUpdated – datetime – The date and time the Task was last updated
• LastUpdatedBy – int – The VRT UserID of the user that last updated the Task
• ActualComplete – datetime – The date and time the Task was actually completed
• CreatedBy – int – The VRT UserID of the user that created the Task
• VisibleToUsers – bit – Indicates if the Task will be visible to Users of the website (0 = not visible, 1 = visible)

**tblTeams**

*Description*

This table stores information about the Teams that have been created on the website. Since Teams can only be assigned to Clients, there is no reason for many-to-many or one-to-many relationships to be allowed, so linkage information is stored in this table as well.

*Fields*

• TeamID – int – The ID of the Team
• TeamName – varchar 50 – The name of the Team
• TeamDesc – text – The description of the Team
• TeamCharter – text – The charter of the Team
• Team Compact – text – The compact of the Team
• ClientID – int – The ID of the Client the Team is linked to, a foreign key to the tblClients table
• CreationDate – datetime – The date and time the Team was created
• **CreatedBy** – int – The VRT UserID of the user that created the Team, a foreign key to the tblUsers table

• **LastUpdated** – datetime – The date and time the Team was last updated

• **LastUpdatedBy** – int – The VRT UserID of the user that last updated the Team, a foreign key to the tblUsers table

• **VisibleToUsers** – bit – Indicates if the Team will be visible to the users of the website (0 = not visible, 1 = visible)

**tblTypes**

**Description**

This table stores information about the different types of items that exist within the website

**Fields**

• **TypeID** – int – The ID of the item type

• **Type** – varchar 50 – The name of the item type

• **InfoProcName** – varchar 100 – The name of the stored procedure used to pull the item type’s information out of the database

• **VarName** – varchar 100 – The name of the variable used by the stored procedure in InfoProcName to pull the item type’s information out of the database

• **ImageName** – varchar 100 – The name of the image file used as an icon to represent this item type

• **SummaryProcName** – varchar 100 – The name of the stored procedure use to pull the item type’s summary information out of the database

• **LabelText** – varchar 50 – The label used when displaying summary information about the item type

**tblUserRoleAssignments**

**Description**
This table stores the linkages between the Users and their roles on the items in the website.

**Fields**

- UserID – int – The VRT UserID of the User being assigned to the Role, a foreign key to the tblUsers table
- RoleAssignmentID – int – The Role that the User is being assigned to, a foreign key to the tblObjectRoleAssignments table

**tblUsers**

**Description**

This table stores information about the Users of the website that is supplemental to that used for the ASP.NET Membership system. This table also establishes the relationship between the Users VRT UserID and their ASP.NET Application UserID.

**Fields**

- UserID – int – The VRT UserID of the User
- FirstName – varchar 100 – The User’s first name
- LastName – varchar 100 – The User’s last name
- Birthday – datetime – The User’s birthday
- Notes – text – Any notes for the User
- City – varchar 100 – The User’s city of residence
- State – varchar 100 – The User’s state or province of residence
- Country – varchar 100 – The User’s country or residence
- PhoneNumber – varchar 100 – The User’s phone number
- TimeZone – varchar 200 – The User’s time zone
• Photo – image – The User’s photo
• PhotoType – varchar 50 – The MIME Type of the User’s photo
• netUserID – uniqueidentifier – The User’s ASP.NET Application UserID
• SkypeName – varchar 100 – The User’s Skype login
• Address1 – varchar 100 – The first line of the User’s mailing address
• Address2 – varchar 100 – The second line of the User’s mailing address
• PostCode – varchar 100 – The User’s mailing Postal Code
• ClientID – int – The ID of the Client the User is assigned to, a foreign key to the tblClients table
• HomePage – varchar 200 – The URL of the User’s home page on the website, this is the page that loads for the User when they first log in, or when they click the “Home” button
• Active – int – Indicates if the User is an active user (0 = inactive, 1 = active)

Stored Procedures

spAddCalEntry

Inputs
• @CreatorName – varchar 100 – The ASP.NET UserName of the user that created the new Calendar Entry
• @EntryName – varchar 50 – The name of the new Calendar Entry
• @EntryDesc – text – The description of the new Calendar Entry
• @ParentID – int – The ID of the item the new Calendar Entry is linked to
• @EventDate – datetime – The date and time the new Calendar Entry is scheduled for
• @EventDuration – varchar 50 – The duration of the new Calendar Entry, stored as an ASP.NET TimeSpan object
• @EventTypeID – int – The ID of the type of Calendar Entry
• @ParentTypeID – int – The ID of the type of Item the new Calendar Entry is linked to
**Outputs**
- @ReturnID – int – The ID of the new Calendar Entry

**Description**
This stored procedure takes the data entered by the user into the Connect New Calendar Entry form on the website and adds it to the database. This stored procedure first creates the new Calendar Entry in the tblCalEntry table, and then adds the linkage information to the tblCalEntryAssignments table. The role assignment information for the new Calendar Entry is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Entry’s creator as the Entry Admin and all other parent item users as Entry Users. Finally if this is the first Calendar Entry being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Calendar Entries are displayed.

**spAddClient**

**Inputs**
- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Client
- @ClientName – varchar 50 – The name of the Client
- @ClientDesc – text – The description of the Client
- @StyleSheet – varchar 50 – The filename of the CSS Style Sheet the Client will use, allows different Clients to have different color schemes
- @ClientMessage – text – A message displayed on the Client’s page, along the lines of a mission statement, or some other Client-wide piece of news
- @Country – varchar 50 – The country where the Client is located
- @BusinessLine – varchar 50 – The business line of the Client

**Outputs**
- None
Description
This stored procedure is called from the website administration interface and takes the information that an administrative user has entered into the Client creation form and enters it into the database. First, the Client’s information is added to the tblClients table, and then the role assignment information is inserted into the tblObjectRoleAssignments table. Finally the creator of the Client is added to the Client Administrator role, since there are no other users for the Client, further role assignments are not necessary.

spAddComment

Inputs
- @CommentData – text – The text of the Discussion Comment entered by the User
- @CreatorName – varchar 100 – The ASP.NET UserName of the User who entered the Comment
- @ParentID – int - The ID of the Discussion the Comment is a part of

Outputs
- @ReturnID – int – The ID of the new Comment

Description
This stored procedure takes the information a User enters into the Connect New Discussion Comment form and inserts it into the database. This stored procedure first creates the new Discussion Comment in the tblComments table. The role assignment information for the new Discussion Comment is then created in the tblObjectRoleAssignments table and the Users of the parent Discussion are then added to the tblUserRoleAssignments table, with the new Comment’s creator as the Comment Admin and all other parent Discussion users as Comment Users.

spAddDeliverable

Inputs
• **@CreatorName** – varchar 100 – The ASP.NET UserName of the User that created the Deliverable

• **@DeliverableName** – varchar 50 – The name of the Deliverable

• **@DeliverableDesc** – text – The description of the Deliverable

• **@DateDue** – datetime – The date and time the Deliverable is due

• **@ParentID** – int – The ID of the Parent Item

• **@ParentTypeID** – int – The ID of the Parent Item’s Type

**Outputs**
• **@ReturnID** – int – The ID of the new Deliverable

**Description**
This stored procedure takes the data entered by the user into the Connect New Deliverable form on the website and adds it to the database. This stored procedure first creates the new Deliverable in the tblDeliverables table, and then adds the linkage information to the tblDeliverableAssignments table. The role assignment information for the new Deliverable is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Deliverable’s creator as the Deliverable Admin and all other parent Item users as Deliverable Users. Finally if this is the first Deliverable being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Deliverables are displayed.

**spAddDepartment**

**Inputs**
• **@CreatorName** – varchar 100 – The ASP.NET UserName of the User that created the Department

• **@DepartmentName** – varchar 50 – The name of the Department

• **@DepartmentDesc** – text – The description of the Department
• @DepartmentLocation – varchar 50 – The geographical location of the Department
• @ParentID – int – The ID of the Region the Department is a part of

Outputs
• @ReturnID – int – The ID of the new Department

Description
This stored procedure takes the data entered by the user into the Connect New Department form on the website and adds it to the database. This stored procedure first creates the new Department in the tblDepartments table. The role assignment information for the new Department is then created in the tblUserRoleAssignments table and the Users of the parent Region are then added to the tblUserRoleAssignments table, with the new Department’s creator as the Department Admin and all other parent Region users as Department Users. Finally if this is the first Department being linked to the parent Region, an entry is added to the tblPageLayout table for the parent Item so the linked Departments are displayed.

spAddDiscussion

Inputs
• @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Discussion
• @DiscussionTopic – varchar 50 – The topic of the Discussion
• @CommentText – text – The text of the first Comment of the Discussion
• @ParentID – int – The ID of the Item the Discussion is linked to
• @ParentTypeID – int – The ID of the type of Item the Discussion is linked to

Outputs
• @ReturnID – int – The ID of the new Discussion

Description
This stored procedure takes the data entered by the user into the Connect New Discussion form on the website and adds it to the database. This stored procedure first creates the new
Discussion in the tblDiscussions table, and then adds the linkage information to the
tblDiscussionAssignments table. The role assignment information for the new Discussion is then
created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to
the tblUserRoleAssignments table, with the new Discussion’s creator as the Discussion Admin
and all other parent Item users as Discussion Users. Finally if this is the first Discussion being
linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the
linked Discussions are displayed.

**spAddEvalCriteria**

**Inputs**
- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the
  Evaluation Criteria
- @Criteria – varchar 50 – The name of the Evaluation Criteria
- @CriteriaDesc – text – The description of the Evaluation Criteria
- @ParentID – int – The ID of the Item the Evaluation Criteria will be linked to
- @ParentTypeID – int – The ID of the type of Item the Evaluation Criteria will be linked to

**Outputs**
- @ReturnID – int – The ID of the new Evaluation Criteria

**Description**
This stored procedure takes the data entered by the user into the Connect New Evaluation
Criteria form on the website and adds it to the database. This stored procedure first creates the
new Evaluation Criteria in the tblEvalCriteria table, and then adds the linkage information to the
tblEvalCriteriaAssignments table. The role assignment information for the new Evaluation
Criteria is then created in the tblObjectRoleAssignments table and the Users of the parent Item
are then added to the tblUserRoleAssignments table, with the new Evaluation Criteria’s creator as
the Evaluation Criteria Admin and all other parent Item users as Evaluation Criteria Users.

Finally if this is the first Evaluation Criteria being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Evaluation Criteria are displayed.

**spAddFactor**

**Inputs**
- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Facilitating Factor
- @FactorName – varchar 50 – The name of the Facilitating Factor
- @FactorText – text – The description of the Facilitating Factor
- @ParentID – int – The ID of the Item the Facilitating Factor will be linked to
- @ParentTypeID – int – The ID of the type of Item the Facilitating Factor will be linked to

**Outputs**
- @ReturnID – int – The ID of the new Facilitating Factor

**Description**
This stored procedure takes the data entered by the user into the Connect New Facilitating Factor form on the website and adds it to the database. This stored procedure first creates the new Facilitating Factor in the tblFactors table. The role assignment information for the new Facilitating Factor is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Facilitating Factor’s creator as the Deliverable Admin and all other parent Item users as Facilitating Factor Users. Finally if this is the first Facilitating Factor being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Facilitating Factors are displayed.

**spAddFile**
**Inputs**

- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the File
- @FileName – varchar 200 – The name of the File
- @FileDesc – text – The description of the File
- @FolderID – int – The ID of the Folder the File will be linked to
- @FileData – image – The actual contents of the file in binary
- @VersionNotes – text – Any notes for this version of the File
- @FileLength – bigint – The length of the File in bytes
- @FileContentType – varchar 100 – The MIME content type of the File
- @IsCurrentVersion – bit – Indicates if this is to be the current version of the file (0 = not current version, 1 = current version)

**Outputs**

- @ReturnID – int – The ID of the new File

**Description**

This stored procedure takes the data entered by the user into the Connect New File form on the website and adds it to the database. This stored procedure first creates the new File in the tblFiles table, adds the file contents and associated information to the tblFileData table, and then adds the linkage information to the tblFileAssignments table. The role assignment information for the new File is then created in the tblObjectRoleAssignments table and the Users of the parent Folder are then added to the tblUserRoleAssignments table, with the new File’s creator as the File Admin and all other parent Folder users as File Users.

**spAddFolder**

**Inputs**

- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Folder
@FolderName – varchar 50 – The name of the Folder
@FolderDesc – text – The description of the Folder
@ParentID – int – The ID of the Item the Folder will be linked to
@ParentTypeID – int – The DI of the type of Item the Folder will be linked to

**Outputs**

@ReturnID – int – The ID of the new Folder

**Description**

This stored procedure takes the data entered by the user into the Connect New Folder form on the website and adds it to the database. This stored procedure first creates the new Folder in the tblFolders table, and then adds the linkage information to the tblFolderAssignments table. The role assignment information for the new Folder is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Folder’s creator as the Folder Admin and all other parent Item users as Folder Users. Finally if this is the first Folder being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Folders are displayed.

**spAddGoal**

**Inputs**

@CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Goal
@GoalName – varchar 50 – The name of the Goal
@GoalDesc – text – The description of the Goal
@EstComplete – datetime – The estimated completion date and time of the Goal
@StartDate – datetime – The date and time work on the Goal will begin
@ParentID – int – The ID of the Item the Goal will be linked to
• @ParentTypeID – int – The ID of the type of Item the Goal will be linked to

**Outputs**
• @ReturnID – int – The ID of the new Goal

**Description**
This stored procedure takes the data entered by the user into the Connect New Goal form on the website and adds it to the database. This stored procedure first creates the new Goal in the tblGoals table. The role assignment information for the new Goal is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Goal’s creator as the Goal Admin and all other parent Item users as Goal Users. Finally if this is the first Goal being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Goals are displayed.

**spAddIssue**

**Inputs**
• @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Issue
• @IssueName – varchar 50 – The name of the Issue
• @IssueText – text – The description of the Issue
• @ParentID – int – The ID of the Item the Issue will be linked to
• @ParentTypeID – int – The ID of the type of Item the Issue will be linked to

**Outputs**
• @ReturnID – int – The ID of the new Issue

**Description**
This stored procedure takes the data entered by the user into the Connect New Issue form on the website and adds it to the database. This stored procedure first creates the new Issue in the tblIssues table. The role assignment information for the new Issue is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the
tblUserRoleAssignments table, with the new Issue’s creator as the Issue Admin and all other parent Item users as Issue Users. Finally if this is the first Issue being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Issues are displayed.

**spAddLesson**

**Inputs**
- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Lesson Learned
- @LessonName – varchar 50 – The name of the Lesson Learned
- @LessonText – text – The description of the Lesson Learned
- @ParentID – int – The ID of the Item the Lesson Learned will be linked to
- @ParentTypeID – int – The ID of the type of Item the Lesson Learned will be linked to

**Outputs**
- @ReturnID – int – The ID of the new Lesson Learned

**Description**
This stored procedure takes the data entered by the user into the Connect New Lesson Learned form on the website and adds it to the database. This stored procedure first creates the new Lesson Learned in the tblLessonsLearned table, and then adds the linkage information to the tblLessonLearnedAssignments table. The role assignment information for the new Lesson Learned is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Lesson Learned’s creator as the Lesson Learned Admin and all other parent Item users as Lesson Learned Users. Finally if this is the first Lesson Learned being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Lessons Learned are displayed.

**spAddMilestone**

**Inputs**
• @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Milestone

• @Milestone – varchar 50 – The name of the Milestone

• @MilestoneDesc – text – The description of the Milestone

• @EstComplete – datetime – The estimated completion date and time of the Milestone

• @ParentID – int – The ID of the Item the Milestone will be assigned to

• @ParentTypeID – int – ID of the type of Item the Milestone will be assigned to

**Outputs**

• @ReturnID – int – The ID of the new Milestone

**Description**

This stored procedure takes the data entered by the user into the Connect New Milestone form on the website and adds it to the database. This stored procedure first creates the new Milestone in the tblMilestones table. The role assignment information for the new Milestone is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Milestone’s creator as the Milestone Admin and all other parent Item users as Milestone Users. Finally if this is the first Milestone being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Milestones are displayed.

**spAddParentUsersToDefaultChildRole**

**Inputs**

• @ParentID – int – The ID of the Parent Item

• @ParentTypeID – int – The ID of the type of the Parent Item

• @ObjectID – int – The ID of the current Item

• @ObjectTypeID – int – The ID of the type of the current Item

**Outputs**
None

**Description**
The stored procedure is not called directly by the User; instead it is called by any of the spAddItem stored procedures. It assigns the Users of the Parent Item, except for the creator of the current Item, to the role of the current Item that is marked as Default in the tblRoles table.

**spAddPoll**

**Inputs**
- @CreatorName – varchar 2000 – The ASP.NET UserName of the User that created the Poll
- @Choices – varchar 2000 – A list of the possible answers to the Poll, separated by pipes “|”
- @SurveyQuestion – varchar 2000 – The question asked by the Poll
- @Anonymous – bit – Indicates if the Poll results will be anonymous (0 = not anonymous, 1= anonymous)
- @ExpirationDate – datetime – The date and time the Poll stops collecting responses
- @ParentID – int – The ID of the Item the Poll will be linked to
- @ParentTypeID – int – The ID of the type of Item the Poll will be linked to
- @StartDate – datetime – The date and time the Poll will start collecting responses
- @DisplayMode – int – Indicates if Poll results will be displayed immediately after a User makes their selection, or only after all Users have made the selection (1 = Immediately, 2 = After all Users)

**Outputs**
- @ReturnID – int – The ID of the new Poll

**Description**
This stored procedure takes the data entered by the user into the Connect New Poll form on the website and adds it to the database. This stored procedure first creates the new Poll in the tblSurveys table, and then adds the linkage information to the tblSurveyAssignments table. The
role assignment information for the new Poll is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Poll’s creator as the Poll Admin and all other parent Item users as Poll Users. Next the possible answers for the Poll are added to the tblPollChoices table. Finally if this is the first Poll being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Polls are displayed.

**spAddPollChoices**

**Inputs**
- @ChoiceList – varchar 2000 – A list of the possible answers to the Poll, separated by pipes “|”
- @SurveyID – int – The ID of the Poll the choices will be possible answers for

**Outputs**
- None

**Description**
This stored procedure is not called directly by the user; instead it is called by spAddPoll and adds the possible responses to a Poll to the tblPollChoices table.

**spAddProject**

**Inputs**
- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Project
- @ProjectName – varchar 50 – The name of the Project
- @ProjectDesc – text – The description of the Project
- @EstComplete – datetime – The date and time the Project should be completed
- @StartDate – datetime – The date and time work should begin on the Project
- @ParentID – int – The ID of the Team the Project is linked to

**Outputs**
Description
This stored procedure takes the data entered by the user into the Connect New Project form on the website and adds it to the database. This stored procedure first creates the new Project in the tblProjects table. The role assignment information for the new Project is then created in the tblObjectRoleAssignments table and the Users of the parent Team are then added to the tblUserRoleAssignments table, with the new Project’s creator as the Project Admin and all other parent Team users as Project Users. Finally if this is the first Project being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Projects are displayed.

spAddRegion

Inputs
- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Region
- @RegionName – varchar 50 – The name of the Region
- @RegionDesc – text – The description of the Region
- @RegionLocation – varchar 100 – The geographical location of the Region
- @ParentID – int – The ID of the Client the Region is linked to

Outputs
- @ReturnID – int – The ID of the new Region

Description
This stored procedure takes the data entered by the user into the Connect New Region form on the website and adds it to the database. This stored procedure first creates the new Region in the tblRegions table. The role assignment information for the new Region is then created in the tblObjectRoleAssignments table and the Users of the parent Client are then added
to the tblUserRoleAssignments table, with the new Region’s creator as the Region Admin and all other parent Client users as Region Users. Finally if this is the first Region being linked to the parent Client, an entry is added to the tblPageLayout table for the parent Item so the linked Regions are displayed.

**spAddResource**

*Inputs*

- @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Resource
- @ResourceName – varchar 50 – The name of the Resource
- @ResourceDesc – text – The description of the Resource
- @Field1Data – text – The data for the Resource’s first field
- @Field2Data – text – The data for the Resource’s second field
- @Field3Data – text – The data for the Resource’s third field
- @Field4Data - text – The data for the Resource’s fourth field
- @Field5Data - text – The data for the Resource’s fifth field
- @Field6Data - text – The data for the Resource’s sixth field
- @Field7Data - text – The data for the Resource’s seventh field
- @Field8Data - text – The data for the Resource’s eight field
- @Field9Data - text – The data for the Resource’s ninth field
- @Field10Data - text – The data for the Resource’s tenth field
- @Field11Data - text – The data for the Resource’s eleventh field
- @Field12Data - text – The data for the Resource’s twelfth field
- @Field13Data - text – The data for the Resource’s thirteenth field
- @Field14Data - text – The data for the Resource’s fourteenth field
• @Field15Data - text – The data for the Resource’s fifteenth field
• @Field16Data - text – The data for the Resource’s sixteenth field
• @Field17Data - text – The data for the Resource’s seventeenth field
• @Field18Data - text – The data for the Resource’s eighteenth field
• @Field19Data - text – The data for the Resource’s nineteenth field
• @Field20Data – text – The data for the Resource’s twentieth field
• @ParentID – int – The ID of the Item the Resource will be linked to
• @ParentTypeID – int – The ID of the type of Item the Resource will be linked to
• @ResourceTypeID – int – The ID of the type of Resource being created

**Outputs**
• @ReturnID – int – The ID of the new Resource

**Description**
This stored procedure takes the data entered by the user into the Connect New Resource form on the website and adds it to the database. This stored procedure first creates the new Resource in the tblResources table, and then adds the linkage information to the tblResourceAssignments table. The role assignment information for the new Resource is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Resource’s creator as the Resource Admin and all other parent Item users as Resource Users. Finally if this is the first Resource being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Resources are displayed.

**spAddTask**

**Inputs**
• @CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Task
@TaskName – varchar 50 – The name of the Task
@TaskDesc – text – The description of the Task
@StartDate – datetime – The date and time work on the Task will begin
@EstComplete – datetime – The date and time the Task should be completed
@ParentID – int – The ID of the Item the Task is linked to
@ParentTypeID – int – The ID of the type of Item the Task is linked to

Outputs
@ReturnID – int – The ID of the new Task

Description
This stored procedure takes the data entered by the user into the Connect New Task form on the website and adds it to the database. This stored procedure first creates the new Task in the tblTasks table. The role assignment information for the new Task is then created in the tblObjectRoleAssignments table and the Users of the parent Item are then added to the tblUserRoleAssignments table, with the new Task’s creator as the Task Admin and all other parent Item users as Task Users. Finally if this is the first Task being linked to the parent Item, an entry is added to the tblPageLayout table for the parent Item so the linked Tasks are displayed.

spAddTeam

Inputs
@CreatorName – varchar 100 – The ASP.NET UserName of the User that created the Team
@TeamName – varchar 50 – The name of the Team
@TeamDesc – text – The description of the Team

Outputs
@ReturnID – int – The ID of the new Team

Description
This stored procedure takes the data entered by the user into the Connect New Team form on the website and adds it to the database. This stored procedure first creates the new Team in the tblTeams table. The role assignment information for the new Team is then created in the tblObjectRoleAssignments table and the Users of the parent Client are then added to the tblUserRoleAssignments table, with the new Team’s creator as the Team Admin and all other parent Client users as Team Users. Finally if this is the first Team being linked to the parent Client, an entry is added to the tblPageLayout table for the parent Item so the linked Teams are displayed.

**spAddUser**

**Inputs**
- @FirstName – varchar 100 – The first name of the User being created
- @LastName – varchar 100 – The last name of the User being created
- @UserName – nvarchar 256 – The ASP.NET UserName of the User being created
- @ClientID – int – The ID of the Client the User will be linked to
- @ASPRoleName – varchar 200 – The name of the ASP.NET Role the User is assigned to

**Outputs**
- None

**Description**

**spAssignDefaultRolesToObject**

**Inputs**
- @ObjectID – int – The ID of the Item the default roles will be assigned to
- @ObjectTypeID – int – The ID of the type of Item the default roles will be assigned to

**Outputs**
- None

**Description**
This stored procedure is not called directly by the User; instead it is called by the spAddItem stored procedures and assigns the roles that are flagged as default for the type of Item to the newly created Item.

**spCheckViewPermission**

**Inputs**
- @UserName – varchar 100 – The UserName of the User we are checking permissions for
- @ObjectID – int – The ID of the Item we are checking permissions for
- @ObjectTypeID – int – The ID of the type of Item we are checking permissions for

**Outputs**
- @Authorized – int – Indicates if the User has permission to view the Item (0 = does not have permission, 1 = has permission)

**Description**
This stored procedure is called each time a page is loaded on the site to make sure that the User has permission to view the Item they are trying to access.

**spCreatorIDFromName**

**Inputs**
- @CreatorName – varchar 200 – The ASP.NET UserName of the User we are getting the VRT UserID for

**Outputs**
- @CreatorID – int – The VRT UserID of the User

**Description**
This stored procedure is only called from within other stored procedures and takes an ASP.NET UserName and finds the VRT UserID that corresponds to it.

**spDeleteUser**

**Inputs**
- @UserName – nvarchar 256 – The ASP.NET UserName of the User being deleted
 Outputs
  • None

Description
This stored procedure is called from the Admin interface and first deletes all roles that the User has been assigned to; it then deletes the User from the tblUsers table.

spEditLayoutColumnSwap

Inputs
  • @SourceAssignmentID – int – The ID of the Page Layout assignment that is being altered

Outputs
  • None

Description
This stored procedure is called when a User moves a Connected Item listing from one column to another. The selected Connected Item listing is assigned to the last position in the other column, and the remaining Connected Item listings in the original column are all moved up a position.

spEditLayoutRowSwap

Inputs
  • @SourceAssignmentID – int – The ID of the source Connected Item Page Layout assignment
  • @DestAssignmentID – int – The ID of the destination Connected Item Page Layout assignment
  • @SourceRow – int – The row of the source Connected Item Page Layout assignment
  • @DestRow – int – The row of the destination Connected Item Page Layout assignment

Outputs
  • None

Description
This stored procedure swaps the position of two Connected Item listings. It places the @DestRow value in the Row field for the Source Page Layout Assignment and the @SourceRow in the Row field for the Destination Page Layout Assignment.

**spEditLayoutShiftUp**

**Inputs**
- @OpenOrder – int – The position that has been left open by the column reassignment
- @ParentID – int – The ID of the Item whose Connected Item Page Layout is being altered
- @ParentTypeID – int – The ID of the type of Item whose Connected Item Page Layout is being altered
- @WhichColumn – int – Which Column of the Connected Item Page Layout is being altered

**Outputs**
- None

**Description**
This stored procedure is called by **spEditLayoutColumnSwap** after the Connected Item Listing has been moved to the other column; this stored procedure moves the remaining Items in the column up one position so there are no gaps in the ordering.

**spGetAAGInfo**

**Inputs**
- @ObjectID – int – The ID of the Item whose At-A-Glance information is being pulled
- @ObjectTypeID – int – The ID of the type of Item whose At-A-Glance information is being pulled
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set with the repeating form of: FieldX, FieldXType, FieldXName. This repeats for each entry in the At-A-Glance table for the type of Item.
Description
This stored procedure pulls the At-A-Glance information for the current Item out of the database.

spGetAllClients

Inputs
• None

Outputs
• An SQL result set containing the ID and Name of all Clients stored in the database

Description
This stored procedure is called by the Admin Interface to populate the list of Clients for User Client assignment.

spGetAllFoldersForObject

Inputs
• @ObjectID – int – The ID of the Item we are getting the list of linked folders for
• @ObjectTypeID – int – The ID of the type of Item we are getting the list of linked folders for
• @UserName – varchar 100 – The ASP.NET UserName of the User

Outputs
• An SQL result set containing a list of all Folders and their Subfolders linked to the Item specified with the ObjectID and ObjectTypeID values.

Description
This stored procedure is called as part of the File edit process is used to populate a list of all possible Folders that the File can be linked to.

spGetAllFoldersForObjectForFolderTree

Inputs
• @ObjectID – int – The ID of the Item we are getting the list of linked folders for
• @ObjectTypeID – int – The ID of the type of Item we are getting the list of linked folders for
• @CurrentFolderID – int – The ID of the Folder currently being edited
• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL Result set containing a list of all Folders linked to the Item specified with the ObjectID and ObjectTypeID values.

**Description**
This stored procedure is called when editing a Folder and is used to populate the list of possible other Folders that the current Folder can be linked to.

**spGetAllUsersAndRolesForObject**

**Inputs**
• @ObjectID – int – The ID of the Item
• @TypeID – int – The ID of the type of Item
• @UserName – varchar 200 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing all Users and their assigned roles for the Item

**Description**
This stored procedure is used to populate the Assigned Users display on the web site.

**spGetCalendarSummaryInfo**

**Inputs**
• @ObjectID – int – The ID of the Item whose linked Calendar Entries are being pulled
• @ObjectTypeID – int – The ID of the type of Item whose linked Calendar Entries are being pulled
• @SortColumn – varchar 100 – The name of the column to be sorted by
• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User
**Outputs**
- An SQL result set containing information about the linked Calendar Entries for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Calendar Entries that are linked to the Item being viewed.

**spGetChildrenForObject**

**Inputs**
- @ObjectTypeID – The ID of the type of Item whose list of allowed children is being pulled

**Outputs**
- An SQL result set listing the types of children that are allowed for the specified ObjectType

**Description**
This stored procedure is used to populate the list of item types displayed in the Connect New Item dialog where the User selects the type of Item they want to Connect.

**spGetClientInfo**

**Inputs**
- @ClientID – int – The ID of the Client
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the specified Client

**Description**
This stored procedure is used to populate the information on the page when viewing a Client on the web site.

**spGetColumnItems**

**Inputs**
- @ObjectID – int – The ID of the Item being displayed
• @ObjectTypeID – int – The ID of the type of Item being displayed
• @DisplayType – int – Controls the information returned by the stored procedure
• @WhichColumn – int – Which column of Connected Items will be returned
• @UserName – varchar 100 – The ASP.NET UserName of the User

Outputs
• If DisplayType = 1: The list of connected Items for a Comment
• If DisplayType = 2: The list of connected Items sorted by Row and then Column
• If DisplayType = 3: The list of connected Items in the column specified by @WhichColumn

Description
This Stored Procedure is used to pull the list of the Types of Items that are linked to the Current Item out of the database. The Connected Item listings are created from this list.

spGetCommentInfo

Inputs
• @CommentID – int – The ID of the Comment
• @UserName – varchar 100 – The ASP.NET UserName of the User

Outputs
• An SQL result set that contains information about the specified Comment

Description
This stored procedure is used to populate the information on the page when viewing a Comment on the web site.

spGetCommentSummaryInfo

Inputs
• @DiscussionID – int – The ID of the Discussion
• @RecsPerPage – int – The number of records to be displayed on a page
• @StartRecord – int – The first record to be displayed on the page
• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set showing the current page of discussion comments

**Description**
This stored procedure pulls a page of discussion comments (specified by the StartRecord and RecsPerPage values). This allows for a fixed number of comments to be displayed at a time to make the page length more manageable for the Users of the site.

**spGetCreatorRoleForObjectType**

**Inputs**
• @ObjectTypeID – int – The type of Object

**Outputs**
• @RoleID – int – The Role ID

**Description**
This stored procedure returns the ID of the Role that is assigned to the specified type of Object.

**spGetCriteriaInfo**

**Inputs**
• @CriteriaID – int – The ID of the Evaluation Criteria
• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set that contains information about the specified Evaluation Criteria

**Description**
This stored procedure is used to populate the information on the page when viewing an Evaluation Criteria on the web site.

**spGetCriteriaSummaryInfo**

**Inputs**
• @ObjectID – int – The ID of the Item whose linked Evaluation Criteria are being pulled
• @ObjectTypeID – int – The ID of the type of Item whose linked Evaluation Criteria are being pulled
• @SortColumn – varchar 100 – The name of the column to be sorted by
• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User

Outputs
• An SQL result set containing information about the linked Evaluation Criteria for the Item

Description
This stored procedure is used to populate the Connected Items listing for Evaluation Criteria that are linked to the Item being viewed.

spGetDeliverableInfo

Inputs
• @DeliverableID – int – The ID of the Deliverable
• @UserName – varchar 100 – The ASP.NET UserName of the User

Outputs
• An SQL result set that contains information about the specified Deliverable

Description
This stored procedure is used to populate the information on the page when viewing a Deliverable on the web site.

spGetDeliverablesSummaryInfo

Inputs
• @ObjectID – int – The ID of the Item whose linked Deliverables are being pulled
• @ObjectTypeID – int – The ID of the type of Item whose linked Deliverables are being pulled
• @SortColumn – varchar 100 – The name of the column to be sorted by
• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Deliverables for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Deliverables that are linked to the Item being viewed.

**spGetDepartmentInfo**

**Inputs**
• @DepartmentID – int – The ID of the Department
• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set that contains information about the specified Department

**Description**
This stored procedure is used to populate the information on the page when viewing a Department on the web site.

**spGetDepartmentSummaryInfo**

**Inputs**
• @ObjectID – int – The ID of the Region whose linked Departments are being pulled
• @ObjectTypeID – int – The ID of the type of Item whose linked Departments are being pulled
• @SortColumn – varchar 100 – The name of the column to be sorted by
• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Departments for the Region
**Description**
This stored procedure is used to populate the Connected Items listing for Departments that are linked to the Region being viewed.

**spGetDiscussionInfo**

**Inputs**
- @DiscussionID – int – The ID of the Discussion
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified Discussion

**Description**
This stored procedure is used to populate the information on the page when viewing a Discussion on the web site.

**spGetDiscussionsSummaryInfo**

**Inputs**
- @ObjectID – int – The ID of the Item whose linked Discussions are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Discussions are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the linked Discussions for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Discussions that are linked to the Item being viewed.

**spGetEntryInfo**
**Inputs**
- @EntryID – int – The ID of the Calendar Entry
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified Calendar Entry

**Description**
This stored procedure is used to populate the information on the page when viewing a Calendar Entry on the web site.

**spGetFactorInfo**

**Inputs**
- @FactorID – int – The ID of the Facilitating Factor
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified Facilitating Factor

**Description**
This stored procedure is used to populate the information on the page when viewing a Facilitating Factor on the web site.

**spGetFactorSummaryInfo**

**Inputs**
- @ObjectID – int – The ID of the Item whose linked Facilitating Factors are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Facilitating Factors are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Facilitating Factors for the Item

**Description**

This stored procedure is used to populate the Connected Items listing for Facilitating Factors that are linked to the Item being viewed.

**spGetFieldNamesForResource**

**Inputs**

- @ResourceID – int – The ID of the Resource whose field names are being pulled

**Outputs**

- An SQL result set containing the list of field names

**Description**

This stored procedure takes a ResourceID and returns the list of field names that correspond to the given resource, determined by the ResourceTypeID field in the tblResources table.

**spGetFileForDownload**

**Inputs**

- @FileDataID – int – The ID of the selected version of the File’s contents
- @FileID – int – The ID of the File itself
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**

- An SQL result set containing the contents of the selected version of the File in binary format and additional information required for the downloading of the File.

**Description**

This stored procedure is called when a User clicks on the download link on the website. It pulls from the database the necessary information so that the file can be reconstructed, it is stored in the database rather than in the server’s file system, and downloaded by the User.
spGetFileInfo

spGetFactorInfo

**Inputs**
- @FileID – int – The ID of the File
- @CurrentVersionID – int – The ID of the version of the File data being pulled
- @SortColumn – varchar 100 – The name of the column to sort by
- @SortOrder – varchar 100 – The direction to sort
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified version of the File
- An SQL result set that contains information about the other versions of the File.

**Description**
This stored procedure is used to populate the information on the page when viewing a Facilitating Factor on the web site.

spGetFileParentFolder

**Inputs**
- @FileID – int – The ID of the File
- @UserName – varchar 100 – The ASP.NET UserID of the User

**Outputs**
- None

**Description**
This stored procedure first pulls the specified File’s parent FolderID from the tblFileAssignments table, and then calls spGetFolderParentForFileTree.

spGetFolderContents

**Inputs**
- @FolderID – int – The ID of the Folder
• @UserName – int – The ASP.NET UserName of the User
• @SortColumn – varchar 100 – The column to be sorted by
• @SortOrder – varchar 100 – The direction to be sorted

Outputs
• An SQL result set containing the contents of the Folder

Description
This stored procedure is not called directly by the User, instead it is called from spGetFolderInfo. The SQL result set returned by this stored procedure contains a listing of both the Files and the subfolders contained within the specified Folder. This result set is sorted according to the values specified in the @SortColumn and @SortOrder values.

spGetFolderInfo

Inputs
• @FolderID – int – The ID of the Folder
• @UserName – varchar 100 – The ASP.NET UserName of the User
• @SortColumn – varchar 100 – The column to be sorted by
• @SortOrder – varchar 100 – The direction to be sorted

Outputs
• An SQL result set containing information about the specified Folder
• The SQL result set returned by spGetFolderContents

Description
This stored procedure returns both the information about the Folder itself, and the listing (both Files and subfolders) of the Folder’s contents.

spGetFolderParent

Inputs
• @FolderID – int – The ID of the Folder
• @UserName – varchar 100 – The ASP.NET UserName of the User

Outputs
• @RootParentID – int – The ID of the Item that is the root of this group of Folders and subfolders.
• @RootParentTypeID – int – The ID of the type of Item that is the root of this group of Folders and subfolders.
• @RootParentName – varchar 2000 – The name of the Item that is the root of this group of Folders and subfolders.
• @ItemParentTypeID – int – The ID of the type of Item that is the parent of the specified Folder.
• @ItemParentID – int – The ID of the Item that is the parent of the specified Folder.
• @ItemParentName – varchar 2000 – The name of the Item that is the parent of the specified Folder.

Description
This stored procedure returns information about the parent Items for a specified Folder. It returns the Name, TypeID and ID of the Folder’s parent Item; it also returns the same information for the root Item of the group of Folders and subfolders that the specified Folder belongs to. This stored procedure calls spGetObjectParentInfoV2 to get information about the specified Folders parent, and spGetAllFoldersForObjectForFolderTree to get all Folders and subfolders in the Folder group.

spGetFolderParentForFileTree

Inputs
• @FolderID – int – The ID of the Folder
• @UserName – varchar 100 – The ASP.NET UserID of the User

Outputs
• The SQL result set created by spGetAllFoldersForObject is returned to the web application.

Description
This stored procedure starts with the specified Folder and checks its parent, if the parent is a Folder it repeats this process until it comes to an Item that is not a Folder. That Item’s information is then passed to spGetGallFoldersForObject to get all Folders and subfolders that are linked to the Item; this information is then passed back to the web application.

**spGetFolderSummaryInfo**

**Inputs**
- @ObjectID – int – The ID of the Item whose linked Folders are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Folders are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the linked Folders for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Folders that are linked to the Item being viewed.

**spGetGoalInfo**

**Inputs**
- @GoalID – int – The ID of the Goal
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified Goal

**Description**
This stored procedure is used to populate the information on the page when viewing a Goal on the web site.
spGetGoalSummaryInfo

Inputs
- @ObjectID – int – The ID of the Item whose linked Goals are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Goals are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

Outputs
- An SQL result set containing information about the linked Goals for the Item

Description
This stored procedure is used to populate the Connected Items listing for Goals that are linked to the Item being viewed.

spGetIssueInfo

Inputs
- @IssueID – int – The ID of the Issue
- @UserName – varchar 100 – The ASP.NET UserName of the User

Outputs
- An SQL result set that contains information about the specified Issue

Description
This stored procedure is used to populate the information on the page when viewing an Issue on the web site.

spGetIssueSummaryInfo

Inputs
- @ObjectID – int – The ID of the Item whose linked Issues are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Issues are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Issues for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Issues that are linked to the Item being viewed.

**spGetItemInfoForEmail**

**Inputs**
• @ObjectID – int – The ID of the Item the e-mail notification is being sent for
• @ObjectTypeID – int – The ID of the type of Item the e-mail notification is being sent for
• @ParentID – int – The ID of the Item that is the parent of the Item the e-mail notification is being sent for
• @ParentTypeID – int – The ID of the type of Item that is the parent of the Item the e-mail notification is being sent for

**Outputs**
• An SQL result set containing the names of the Item and its parent Item
• An SQL result set containing the e-mail address of all Users that are assigned to the Item

**Description**
This stored procedure retrieve the information needed for sending e-mail notifications about changes to the web site.

**spGetLessonInfo**

**Inputs**
• @LessonID – int – The ID of the Lesson Learned
• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set that contains information about the specified Lesson Learned

**Description**
This stored procedure is used to populate the information on the page when viewing a Lesson Learned on the web site.

**spGetLessonSummaryInfo**

**Inputs**
• @ObjectID – int – The ID of the Item whose linked Lessons Learned are being pulled

• @ObjectTypeID – int – The ID of the type of Item whose linked Lessons Learned are being pulled

• @SortColumn – varchar 100 – The name of the column to be sorted by

• @SortOrder – varchar 100 – The direction of the sort

• @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Lessons Learned for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Lessons Learned that are linked to the Item being viewed.

**spGetMilestoneInfo**

**Inputs**
• @MilestoneID – int – The ID of the Milestone

• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set that contains information about the specified Milestone

**Description**
This stored procedure is used to populate the information on the page when viewing a Milestone on the web site.
**spGetMilestoneSummaryInfo**

**Inputs**
- @ObjectID – int – The ID of the Item whose linked Milestones are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Milestones are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the linked Milestones for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Milestones that are linked to the Item being viewed.

**spGetObjectChildren**

**Inputs**
- @ObjectID – int – The ID of the Item whose children are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose children are being pulled
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the child Items of the specified Item.

**Description**
This stored procedure is used to populate the Item Overview pages which are used to show upcoming due dates for Items on the website. The result set is sorted by if the Item has a date associated with it (Calendar Entry, Milestone, Project, etc…) and then sorted by Name.

**spGetObjectParentInfoV2**

**Inputs**
@ObjectID – int – The ID of the Item whose parent information is being pulled

@ObjectTypeID – int – The ID of the type of Item whose parent information is being pulled

**Outputs**

- @ParentID – int – The ID of the Item that is the parent of the specified Item
- @ParentTypeID – int – The ID of the type of Item that is the parent of the specified Item
- @ParentName – varchar 2000 – The name of the Item that is the parent of the specified Item

**Description**

This stored procedure takes the ID and TypeID of the current Item and returns the ID, TypeID and name of the specified Item’s parent Item.

**spGetOverviewInfo**

**Inputs**

- @ObjectID – int – The ID of the Item whose overview information is being pulled
- @ObjectTypeID – int – The ID of the type of Item whose overview information is being pulled

**Outputs**

This stored procedure is called while building the Item Overview on the web site. The name and any associated date information is returned to the web application to populate the Overview list or calendar.

**Description**

**spGetPollChoices**

**Inputs**

- @SurveyID – int – The ID of the Poll whose choices are being pulled

**Outputs**

- An SQL result set containing the choices for the specified Poll

**Description**
This stored procedure returns the Choices, or possible answers for the specified Poll.

**spGetPollInfo**

**Inputs**
- @SurveyID – int – The ID of the Poll
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified Poll

**Description**
This stored procedure is used to populate the information on the page when viewing a Poll on the web site.

**spGetPollResults**

**Inputs**
- @SurveyID – int – The ID of the Poll whose results are being pulled
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- If the Poll has not opened yet a message to that effect is returned
- If not all Users have submitted results and the DisplayMode is set so that results are not displayed until all Users have submitted result a message to that effect is returned, along with a list of the Users who have not yet responded
- If the User has not submitted their vote a message to that effect is returned
- Otherwise, the results of the Poll are returned, as is the total number of votes received, the number of votes for each Choice and each User’s vote.

**Description**
This stored procedure is used to display the results of the Poll, or an appropriate status message.

**spGetPollSummaryInfo**

**Inputs**
• @ObjectID – int – The ID of the Item whose linked Polls are being pulled
• @ObjectTypeID – int – The ID of the type of Item whose linked Polls are being pulled
• @SortColumn – varchar 100 – The name of the column to be sorted by
• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Polls for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Polls that are linked to the Item being viewed.

**spGetProjectInfo**

**Inputs**
• @ProjectID – int – The ID of the Issue

• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set that contains information about the specified Project

**Description**
This stored procedure is used to populate the information on the page when viewing a Project on the web site.

**spGetProjectsSummaryInfo**

**Inputs**
• @ObjectID – int – The ID of the Team whose linked Projects are being pulled

• @ObjectTypeID – int – The ID of the type of Item whose linked Projects are being pulled

• @SortColumn – varchar 100 – The name of the column to be sorted by

• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Projects for the Team

**Description**
This stored procedure is used to populate the Connected Items listing for Projects that are linked to the Team being viewed.

**spGetRegionInfo**

**Inputs**
• @RegionID – int – The ID of the Region
• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set that contains information about the specified Region

**Description**
This stored procedure is used to populate the information on the page when viewing a Region on the web site.

**spGetRegionSummaryInfo**

**Inputs**
• @ObjectID – int – The ID of the Client whose linked Regions are being pulled
• @ObjectTypeID – int – The ID of the type of Item whose linked Regions are being pulled
• @SortColumn – varchar 100 – The name of the column to be sorted by
• @SortOrder – varchar 100 – The direction of the sort
• @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
• An SQL result set containing information about the linked Regions for the Client

**Description**
This stored procedure is used to populate the Connected Items listing for Regions that are linked to the Client being viewed.

**spGetResourceInfo**

**Inputs**
- @ResourceID – int – The ID of the Resource
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified Resource

**Description**
This stored procedure is used to populate the information on the page when viewing a Resource on the web site.

**spGetResourceSummaryInfo**

**Inputs**
- @ObjectID – int – The ID of the Item whose linked Resources are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Resources are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the linked Resources for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Resources that are linked to the Item being viewed.

**spGetResourceTypeInfo**

**Inputs**
• @ResourceTypeID – int – The ID of the type of Resource whose information is being pulled

**Outputs**

• An SQL result set containing the Name and Type of each field of the Resource

**Description**

The web site allows for customization of resources by storing name and data type information separately from the resource itself so multiple resources of different types can be stored in the same table. The stored procedure returns the name and type information for input validation or field labelling when displaying information about the specified Resource.

**spGetRolesForObject**

**Inputs**

• @ObjectID – int – The ID of the Item whose Roles are being pulled

• @ObjectTypeID – int – The ID of the type of Item whose Roles are being pulled

**Outputs**

• An SQL result set containing the list of Roles for an Item

**Description**

This stored procedure returns the list of Roles for a given Item; this is used to populate the Roles list on the User Permissions form.

**spGetTaskInfo**

**Inputs**

• @TaskID – int – The ID of the Task

• @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**

• An SQL result set that contains information about the specified Task

**Description**

This stored procedure is used to populate the information on the page when viewing a Task on the web site.
spGetTaskSummaryInfo

**Inputs**
- @ObjectID – int – The ID of the Item whose linked Tasks are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Tasks are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the linked Tasks for the Item

**Description**
This stored procedure is used to populate the Connected Items listing for Tasks that are linked to the Item being viewed.

spGetTeamInfo

**Inputs**
- @TeamID – int – The ID of the Team
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified Team

**Description**
This stored procedure is used to populate the information on the page when viewing a Team on the web site.

spGetTeamSummaryInfo

**Inputs**
- @ObjectID – int – The ID of the Client whose linked Teams are being pulled
- @ObjectTypeID – int – The ID of the type of Item whose linked Teams are being pulled
- @SortColumn – varchar 100 – The name of the column to be sorted by
- @SortOrder – varchar 100 – The direction of the sort
- @UserName – varchar 2000 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set containing information about the linked Teams for the Client

**Description**
This stored procedure is used to populate the Connected Items listing for Teams that are linked to the Client being viewed.

**spGetUserDefaultPage**

**Inputs**
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- If the User’s home page is specified in the database, an SQL Result set containing the URL of their custom home page, their VRT UserID, and their Time Zone is returned
- If the User’s home page is not specified in the database, an SQL Result set containing the URL of their Organization’s page, their VRT UserID, and their Time Zone is returned

**Description**
This stored procedure is called both when the User first logs in, and when they click the “Home” link on the website. It is used to redirect the User to their chosen default page and also to populate the TimeZone value of the User’s ASP.NET Profile.

**spGetUserInfo**

**Inputs**
- @UserID – int – The ID of the User whose information is being accessed
- @UserName – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- An SQL result set that contains information about the specified User

**Description**
This stored procedure is used to populate the information on the page when viewing a User on the web site.

**spGetUserRoleAssignmentForObject**

**Inputs**
- @UserID – int – The VRT UserID of the User
- @ObjectID – int – The ID of the Item
- @ObjectTypeID – int – The ID of the type of Item

**Outputs**
- An SQL Result set containing the RoleAssignmentID for the User on the specified Item

**Description**
This stored procedure is used to pull the RoleAssignment ID of the Role that the User is assigned to on the specified Item. This information is used to automatically select the User’s Role in the drop down list showing the different roles for the specified Item on the edit User Permission form.

**spInsertPollResults**

**Inputs**
- @Data – varchar 2000 – The option the User selected when taking the Poll
- @UserName – varchar 100 – The ASP.NET UserName of the User
- @SurveyID – int – The ID of the Poll the User took

**Outputs**
- @Anonymous – bit – Indicates if the Poll results are anonymous or not (0 = not anonymous, 1 = anonymous)

**Description**
This stored procedure stores the User’s selection in a Poll in the database. @Anonymous is returned to control the display of results in the web application after the User has submitted their response.
spIsSelf

**Inputs**
- @name – varchar 100 – The ASP.NET UserName of the User

**Outputs**
- @id – int – The VRT UserID of the User

**Description**
This stored procedure takes a User’s ASP.NET UserName and returns their VRT UserID. This is used when viewing the profile of a User on the web site. If the UserID of the current User matches the UserID of the User whose profile they are viewing, the change password dialog is shown.

spSetUserHomePage

**Inputs**
- @UserName – varchar 100 – The ASP.NET UserName of the User
- @HomePage – varchar 200 – The URL of the page the User has selected to be their home/default page.

**Outputs**
- None

**Description**
This stored procedure sets the User’s default page in the database. It is called when the User clicks the “Set As Home” link on the web site.

spUpdateClientInfo

**Inputs**
- @ClientID – int – The ID of the Client being edited
- @UserName – varchar 100 – The ASP.NET UserName of the User
- @ClientName – varchar 50 – The name of the Client
- @ClientDesc – text – The description of the Client
• @StyleSheet – varchar 50 – The filename of the CSS Style Sheet the Client will use, allows different Clients to have different color schemes

• @ClientMessage – text – A message displayed on the Client’s page, along the lines of a mission statement, or some other Client-wide piece of news

• @Country – varchar 50 – The country where the Client is located

• @BusinessLine – varchar 50 – The business line of the Client

**Outputs**

• None

**Description**

This stored procedure takes the information entered into the Edit Client form and submits it to the database.

**spUpdateCommentInfo**

**Inputs**

• @CommentID – int – The ID of the Discussion Comment being edited

• @CommentData – text – The text of the Discussion Comment

• @VisibleToUsers – int – Indicates if the Discussion Comment will be visible to the Users of the web site (0 = not visible, 1 = visible)

• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Discussion Comment

**Outputs**

• None

**Description**

This stored procedure takes the information entered into the Edit Comment form and submits it into the database.

**spUpdateCriteriaInfo**

**Inputs**

• @CriteriaID – int – The ID of the Evaluation Criteria being edited

• @CriteriaName – varchar 100 – The name of the Evaluation Criteria
• @CriteriaDesc – text – The description of the Evaluation Criteria
• @VisibleToUsers – int – Indicates if the Evaluation Criteria will be visible to the Users of the web site (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Discussion Comment

Outputs
• None

Description
This stored procedure takes the information entered into the Edit Evaluation Criteria form and submits it into the database.

spUpdateDeliverableInfo

Inputs
• @DeliverableID – int – The ID of the Deliverable being edited
• @Status – int – Indicates the status of the Deliverable, value is selected from the entries in the tblStatus table
• @DeliverableName – varchar 100 – The name of the Deliverable
• @DeliverableDesc – text – The description of the Deliverable
• @DateDue – datetime – The date and time the Deliverable is due
• @VisibleToUsers – int – Indicates if the Deliverable will be visible to the Users of the web site (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Deliverable

Outputs
• None

Description
This stored procedure takes the information entered into the Edit Deliverable form and submits it into the database.
spUpdateDepartmentInfo

*Inputs*
- @DepartmentID – int – The ID of the Department being edited
- @DepartmentName – varchar 100 – The name of the Department
- @DepartmentDesc – text – The description of the Department
- @DepartmentLocation – varchar 100 – The geographical location of the Department
- @VisibleToUsers – int – Indicates if the Department will be visible to the Users of the website (0 = not visible, 1 = visible)
- @UserName – varchar 100 – The ASP.NET UserName of the User editing the Department

*Outputs*
- None

*Description*
This stored procedure takes the information entered into the Edit Department form and submits it into the database.

spUpdateDiscussionInfo

*Inputs*
- @DiscussionID – int – The ID of the Discussion being edited
- @DiscussionTopic – varchar 100 – The topic of the Discussion
- @VisibleToUsers – int – Indicates if the Discussion will be visible to the Users of the website (0 = not visible, 1 = visible)
- @UserName – varchar 100 – The ASP.NET UserName of the User editing the Discussion

*Outputs*
- None

*Description*
This stored procedure takes the information entered into the Edit Discussion form and submits it into the database.
spUpdateEntryInfo

**Inputs**
- @EntryID – int – The ID of the Calendar Entry being edited
- @EntryName – varchar 100 – The name of the Calendar Entry
- @EntryDesc – text – The description of the Calendar Entry
- @EventDate – datetime – The date and time of the Calendar Entry
- @EventDuration – varchar 50 – The duration of the Calendar Entry, stored as a .NET TimeSpan variable
- @EntryType – int – The type of Calendar Entry, value is selected from the tblCalEntryTypes table
- @VisibleToUsers – int – Indicates if the Calendar Entry will be visible to the Users of the web site (0 = not visible, 1 = visible)
- @UserName – varchar 100 – The ASP.NET UserName of the User editing the Calendar Entry

**Outputs**
- None

**Description**
This stored procedure takes the information entered into the Edit Calendar Entry form and submits it into the database.

spUpdateFactorInfo

**Inputs**
- @FactorID – int – The ID of the Facilitating Factor being edited
- @FactorName – varchar 100 – The name of the Facilitating Factor
- @FactorDesc – text – The description of the Facilitating Factor
- @VisibleToUsers – int – Indicates if the Facilitating Factor will be visible to the Users of the web site (0 = not visible, 1 = visible)
- @UserName – varchar 100 – The ASP.NET UserName of the User editing the Facilitating Factor
Outputs

• None

Description
This stored procedure takes the information entered into the Edit Facilitating Factor form and submits it into the database.

spUpdateFileInfo

Inputs

• @FileID – int – The ID of the File being edited
• @FileDesc – text – The Description of the File being edited
• @FolderID – int – The ID of the Folder the File is linked to
• @FileData – image – The contents of the File in binary format
• @FileLength – bigint – The size of the File in bytes
• @FileContentType – varchar 100 – The MIME content type of the File
• @NewVersionNotes – text – Notes for the new version of the File
• @CurrentVersionNotes – text – Notes for the current version of the File
• @CurrentVersionID – int – The ID of the FileData for the current version of the File
• @VisibleToUsers – int – Indicates if the File will be visible to the Users of the web site (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the File

Outputs

• @NewVersionID – int – The ID of the FileData for the new version of the File

Description
This stored procedure is used to update information about the Files stored in the database. First it updates the information contained in the tblFiles table, and then it updates any changes made to the current version of the File. If a new version of the File is being uploaded, the new
version’s information is inserted into the tblFileData table and the CurrentVersion flag is set for the new file data. Finally any changes are made to the Folder the File is linked to.

**spUpdateFolderInfo**

**Inputs**
- @FolderID – int – The ID of the Folder being edited
- @FolderName – varchar 100 – The name of the Folder
- @FolderDesc – text – The description of the Folder
- @ParentID – int – The ID of the Item the Folder is linked to
- @ParentTypeID – int – The ID of the type of Item the Folder is linked to
- @VisibleToUsers – int – Indicates if the Folder will be visible to the Users of the web site (0 = not visible, 1 = visible)
- @UserName – varchar 100 – The ASP.NET UserName of the User editing the Folder

**Outputs**
- None

**Description**
This stored procedure is used to update information about the Folders stored in the database. First it updates the information about the Folder stored in the tblFolders table, and then it updates the Folders assignment information in the tblFolderAssignments table.

**spUpdateGoalInfo**

**Inputs**
- @GoalID – int – The ID of the Goal being edited
- @GoalName – varchar 100 – The name of the Goal
- @GoalDesc – text – The description of the Goal
- @StartDate – datetime – The date and time work on the Goal will begin
- @EstComplete – datetime – The date and time work on the Goal should be complete
• @ActualComplete – datetime – The date and time work on the Goal was actually completed
• @Status – int – The status of the Goal, values are selected from the tblStatuses table
• @VisibleToUsers – int – Indicates if the Goal will be visible to the Users of the web site (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Goal

**Outputs**
• None

**Description**
This stored procedure takes the information entered into the Edit Goal form and submits it into the database.

### spUpdateIssueInfo

**Inputs**
• @IssueID – int – The ID of the Issue being edited
• @IssueName – varchar 100 – The name of the Issue
• @IssueDesc – text – The description of the Issue
• @VisibleToUsers – int – Indicates if the Issue will be visible to the Users of the web site (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Issue

**Outputs**
• None

**Description**
This stored procedure takes the information entered into the Edit Issue form and submits it into the database.

### spUpdateLessonInfo

**Inputs**
• @LessonID – int – The ID of the Lesson Learned being edited
• @LessonName – varchar 100 – The name of the Lesson Learned
• @LessonDesc – text – The description of the Lesson Learned
• @VisibleToUsers – int – Indicates if the Lesson Learned will be visible to the Users of the web site (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Lesson Learned

Outputs
• None

Description
This stored procedure takes the information entered into the Edit Lesson Learned form and submits it into the database.

spUpdateMilestoneInfo

Inputs
• @Milestone ID – int – The ID of the Milestone being edited
• @MilestoneName – varchar 100 – The name of the Milestone
• @MilestoneDesc – text – The description of the Milestone
• @EstComplete – datetime – The date and time the Milestone is estimated to be completed
• @Status – int – The status of the Milestone, values are taken from the tblStatuses table
• @ActualComplete – datetime – The date and time the Milestone was actually completed
• @VisibleToUsers – int – Indicates if the Milestone will be visible to the Users of the web site (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Milestone

Outputs
• None

Description
This stored procedure takes the information entered into the Edit Milestone form and submits it into the database.

**spUpdatePollInfo**

**Inputs**
- @SurveyID – int – The ID of the Poll being edited
- @PollQuestion – varchar 2000 – The question being asked by the Poll
- @Choices – varchar 2000 – The list of options for the Poll
- @Anonymous – bit – Indicates if the Poll will be anonymous or not (0 = not anonymous, 1 = anonymous)
- @DisplayMode – int – The display mode for the Poll, values are taken from the tblPollDisplayModes table
- @StartDate – datetime – The date and time the Poll will begin collecting responses
- @ExpirationDate – datetime – The Date and time the Poll will stop collecting responses
- @VisibleToUsers – int – Indicates if the Poll will be visible to the Users of the web site (0 = not visible, 1 = visible)
- @UserName – varchar 100 – The ASP.NET UserName of the User editing the Poll

**Outputs**
- None

**Description**
This stored procedure takes the information entered into the Edit Poll form and submits it into the database.

**spUpdateProjectInfo**

**Inputs**
- @ProjectID – int – The ID of the Project being edited
- @ProjectName – varchar 100 – The name of the Project
- @ProjectDesc – text – The description of the Project
• @StartDate – datetime – The date and time work will begin on the Project
• @EstComplete – datetime – The date and time work should be completed on the Project
• @ActualComplete – datetime – The date and time work on the Project was actually completed
• @Status – int – The status of the Project, values are taken from the tblStatuses table
• @VisibleToUsers – int – Indicates if the Project will be visible to the Users of the website (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Project

Outputs
• None

Description
This stored procedure takes the information entered into the Edit Project form and submits it into the database.

spUpdateRegionInfo

Inputs
• @RegionID – int – The ID of the Region being edited
• @RegionName – varchar 100 – The name of the Region
• @RegionDesc – text – The description of the Region
• @RegionLoc – varchar 100 – The geographical location of the Region
• @VisibleToUsers – int – Indicates if the Region will be visible to the Users of the website (0 = not visible, 1 = visible)
• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Region

Outputs
• None

Description
This stored procedure takes the information entered into the Edit Region form and submits it into the database.
**spUpdateResourceInfo**

*Inputs*

- @ResourceID – int – The ID of the Resource being edited
- @ResourceName – varchar 100 – The name of the Resource
- @ResourceDesc – text – The description of the Resource
- @Field1Data – text – The data for the first field of the Resource
- @Field2Data – text – The data for the second field of the Resource
- @Field3Data – text – The data for the third field of the Resource
- @Field4Data – text – The data for the fourth field of the Resource
- @Field5Data – text – The data for the fifth field of the Resource
- @Field6Data – text – The data for the sixth field of the Resource
- @Field7Data – text – The data for the seventh field of the Resource
- @Field8Data – text – The data for the eighth field of the Resource
- @Field9Data – text – The data for the ninth field of the Resource
- @Field10Data – text – The data for the tenth field of the Resource
- @Field11Data – text – The data for the eleventh field of the Resource
- @Field12Data – text – The data for the twelfth field of the Resource
- @Field13Data – text – The data for the thirteenth field of the Resource
- @Field14Data – text – The data for the fourteenth field of the Resource
- @Field15Data – text – The data for the fifteenth field of the Resource
- @Field16Data – text – The data for the sixteenth field of the Resource
- @Field17Data – text – The data for the seventeenth field of the Resource
- @Field18Data – text – The data for the eighteenth field of the Resource
- @Field19Data – text – The data for the nineteenth field of the Resource
• @Field20Data – text – The data for the twentieth field of the Resource

• @VisibleToUsers – int – Indicates if the Resource will be visible to the Users of the website (0 = not visible, 1 = visible)

• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Resource

**Outputs**

• None

**Description**

This stored procedure takes the information entered into the Edit Resource form and submits it into the database.

**spUpdateTaskInfo**

**Inputs**

• @TaskID – int – The ID of the Task being edited

• @TaskName – varchar 100 – The name of the Task

• @TaskDesc – text – The description of the Task

• @StartDate – datetime – The date and time work on the Task will begin

• @EstComplete – datetime – The date and time work on the Task should be complete

• @ActualComplete – datetime – The date and time work on the Task was actually completed

• @Status – int – The status of the Task, values are taken from the tblStatuses table

• @VisibleToUsers – int – Indicates if the Task will be visible to the Users of the website (0 = not visible, 1 = visible)

• @UserName – varchar 100 – The ASP.NET UserName of the User editing the Task

**Outputs**

• None

**Description**

This stored procedure takes the information entered into the Edit Task form and submits it into the database.
spUpdateTeamInfo

**Inputs**
- @TeamID – int – The ID of the Team being edited
- @TeamName – varchar 100 – The name of the Team
- @TeamDesc – text – The description of the Team
- @TeamCharter – text – The charter of the Team
- @TeamCompact – text – The compact of the Team
- @VisibleToUsers – int – Indicates if the Team will be visible to the Users of the web site (0 = not visible, 1 = visible)
- @UserName – varchar 100 – The ASP.NET UserName of the User editing the Team

**Outputs**
- None

**Description**
This stored procedure takes the information entered into the Edit Team form and submits it into the database.

spUpdateUserInfo

**Inputs**
- @UserID – int – The VRT UserID of the User being edited
- @FirstName – varchar 100 – The first name of the User
- @LastName – varchar 100 – The last name of the User
- @Notes – text – Any notes for the User
- @Address1 – varchar 100 – The first line of the User’s address
- @Address2 – varchar 100 – The second line of the User’s address
- @State – varchar 100 – The state of province the User resides in
- @PostCode – varchar 100 – The postal code for the User
• @Country – varchar 100 – The country of the User
• @PhoneNumber – varchar 100 – The phone number for the User
• @Birthday – datetime – The birthday of the User
• @City – varchar 100 – The city the User resides in
• @SkypeName – varchar 100 – The User’s Skype login
• @TimeZone – varchar 200 – The User’s Time Zone, values are taken from the server’s list of Time Zones stored in the registry
• @FileData – image – The User’s picture stored as binary data
• @FileType – varchar 50 – The MIME content type of the User’s picture

**Outputs**
- None

**Description**
This stored procedure takes the information entered into the Edit User form and submits it into the database.

**spUpdateUserObjectRole**

**Inputs**
- @UserID – int – The User whose role is being edited
- @ObjectID – int – The ID of the Item the role is being edited for
- @ObjectTypeID – int – the ID of the type of Item the role is being edited for
- @RoleAssignmentID – int – The ID of the role assignment being edited

**Outputs**
- None

**Description**
This stored procedure either inserts the new User role assignment into the database, or updates the User’s current role assignment with their new role on the Item.
Works Cited


