CHAPTER 3
ENHANCING E-LEARNING THROUGH VIRTUAL REALITY

It is believed by many educationalists that interactivity which allows “learning by doing” arouses interest and generates motivation providing a more engaging experience for the learner [129, 130]. The much-used quote from Lao-Tse written in the 5th century BC sums this up:

“If you tell me, I will listen
If you show me, I will see
If you let me experience, I will learn.”

The Web-Based Education Commission (WBEC) on 19-12-2000 has reported to the President and Congress of the United States on The Power of the Internet for Learning as follows: “We have used the Internet in a narrow fashion, like vast text books or lectures on-line, instead of exploring its interactive potential.” Cultural heritage, education, and entertainment are notable examples of application domains that can largely benefit from the use of Virtual Reality (VR) technologies. This chapter explores the use of VR technologies in three dimensional e-learning through Virtual Reality Modeling Language (VRML).

A virtual tour on the web has become fashionable. Computer simulations are used as educational tools now-a-days to impart knowledge by understanding through interaction. Earlier days they were mainly used in the field of aviation and medical imaging. VR technology provides almost real or believable experiences in a virtual manner. It allows students to explore and manipulate computer generated real objects in different forms ranging from 3-dimensional geometric shapes to highly interactive experiments. As academics become more skilled in tailoring or creating their own learning environments, discipline specific e-learning environments will flourish. Learners are connected with educational resources through the Internet and the World Wide Web in particular. Learning becomes a task-driven process in a Web-based educational system with the hypermedia form of educational material. Much research and technological work has been done in order to support and/or to offer e-learning services.

E-learning offers opportunities to create materials that not only transmit information but also engage the learning in the culture of the discipline. The strength of VR is the opportunity for a student to observe and maneuver normally inaccessible
objects, variables, and processes in a real-time environment. “VR bridges the gap between the concrete world of nature and the abstract world of concepts and models.”[131] Students understand the concepts better than the conventional study because they learn by doing rather than by reading. They also can repeatedly revisit the environment which helps in their learning process. “There is simply no other way to engage students as virtual reality can” [132]. It motivates learners to explore alternative navigational paths through the domain knowledge and from different resources around the globe. An effective e-learning system, which provides valuable learning experience, should combine pedagogy, learning content and community features, in an effective way [133].

3.1 Introduction

The current modern scenario has several technological break-through in which the Internet has ingrained itself into our culture. Internet is used by learners throughout the world to gain new knowledge and skills. Advances in information and communication technology (ICT) offer incredible promises for improving teaching and learning process. Easy to access and use web interaction media support collaborative knowledge creation and sharing. Most of the information we access through the web are static in nature with text and two-dimensional pictures and tables. With the advent of new technologies, web pages can be created that extend beyond two-dimension. These technologies impact the way we think, learn, and interact and has a tremendous potential in educational domain. They enable students to navigate through a three-dimensional computer environment and engage in educational activities thus rapidly acquire knowledge that is customized to their learning requirements.

There are several ways in which VR technology is expected to assist learning. Most importantly it allows students to visualize abstract concepts, to observe events at atomic or planetary scales, and to visit environments and interact with events that distance, time, or safety factor make unavailable. The types of activities supported by this technology promote current educational thinking that students are better able to master, retain, and generalize new knowledge when they are actively involved in constructing that knowledge in a hand on learning environment. VR can offer an effective medium for enhancing certain skills. It provides ways to use 3D visualizations with which the user can interact. Users can exhibit noticeable individual differences in their interaction styles, and abilities to interact with the 3D
environment. Empirical evidence on the effectiveness of virtual environments for promoting learning of rich subject matters is limited.

A VLE can be dynamically adapted to an individual learner to better support the actual learning process and increase the usability. Dynamic contents can be created or adapted in response to interactions of end-users. Adaptation can happen for navigation and interaction in order to help the users in finding and using information more efficiently. Changes in the user interests, knowledge level or communication characteristics might imply the need to adapt the contents to the new session. Values changed are stored in parameters. After updating, the system must readapt the contents and interaction mechanisms enabled for each client in an efficient, user friendly and flexible way.

New mechanisms are required to dynamically customize the associated information to some external parameters. VRML worlds available on the web have so far been relatively limited in terms of scalability and adaptability. The PROTO node in VRML is capable of handling all the necessary adaptations providing a mechanism to interact between objects. When a feature has to be changed while being inside a 3D world and to see the effect immediately, PROTO that can undergo interactive changes is used. Each time the parameters change, the VRML code has to be replaced and re-downloaded at the client site to achieve the adaptation of the rendered scene.

The knowledge related to computer graphics principles, virtual reality, web3D and other advanced information visualization tools and techniques has gained growing accessibility over the last decade due to low cost, high quality hardware and software facilities. Three-dimensional, computer-generated, realistic-looking environments are emerging as important new technologies for education. The design of rich virtual experiences enables learners to access graphical virtual context with computerized agents. VR has the power to facilitate learning. VR environments can be designed for students to figure out their learning objects. It ranges from simple environments presented on a desktop computer to fully immersive multisensory environments experienced through complex headgear and bodysuits. In all of its manifestations, VR is basically a way of simulating or replicating an environment and giving the user a sense of being there, taking control, and personally interacting with that environment with his/her own body.
3.2 Components of a Virtual Environment

It is a challenging task to provide a meaningful interactive virtual environment that is responsive to learners allowing them to actively participate in the learning process. The content designer must have a prior knowledge of the components of a virtual environment.

**Scene:** The scene corresponds to the 3D space in which the objects are located. It contains lights, viewpoints and cameras.

**Objects:** Objects are usually 3D, but there can also be 2D objects in the 3D space. They have a visual representation with color and material properties, a size, a position in the world, and an orientation.

**Behaviors:** The objects may have behaviors. Behaviors may reflect real life behaviors. For instance, objects may be able to move, rotate, change size, transform, and so on. Navigation through the 3D space is achieved by the behavior of the user’s activity. In general, the 3D space also contains objects without behavior. Behaviors are used to make the VLE dynamic, i.e. to create environments where objects are active by performing some behaviors. VRML comes with a range of sensors (e.g., visibility, proximity, touch,…) that can be used in order to monitor user’s behavior in the 3D world.

**User Interaction:** In a VE, the user is able to interact with the objects. For example, a user may pick up an object and drag it to some other place in the space (if the object is moveable). User interaction can also trigger behavior, e.g., clicking on an object may start a behavior. User interaction may be achieved by means of a regular mouse and keyboard.

**Communication:** Remote users can interact with each other, e.g., talk or chat to each other or perform activities together. A virtual environment can be shared among multiple users, all users with the same view of the environment.

**Sound:** Sound can be important in simulations, to enhance the feeling of reality or simply to simulate some sound. Sound/speech can also be used as an instruction and feedback mechanism during the learning process.
Objects populating the VLE have a visual appearance in terms of geometry (shape) and material properties (colors and textures). To enhance the usability of a VLE for a learner, the visual appearance of an object may have to be changed during the lifetime of the VLE. The recent trends over the web on 3D visualization are the major motivational objective for this research. Navigating in 3D virtual reality means that the user has the ability to change the viewing conditions. As virtual reality has continued to develop, non-immersive or desktop VR applications have developed and are making inroads into educational training and development. Potentials of VR applications have been realized in many disciplines. VR with its increasing dynamic, interactive and experiential characteristics allows simulating real environments with various degrees of realism. On-screen representation of real or imaginary world in any context is necessary for the design, simulation, engineering and visualization activities. The foundation of VR is to make the virtual worlds interactive with feedback to users. Interactivity results in deeper learning because learners can hypothesize to test their understanding, learn by mistakes and make sense of the unexpected [134].

Desktop VR focuses on mouse-controlled navigation through a 3D environment on a graphics monitor under computer control. Today the wide-spread availability of sophisticated computer graphics software and reasonably priced computers with high-end graphics hardware components have placed the world of virtual reality on everyone's desktop. Desktop VR tend to be more appropriate in terms of affordability and possibility of wide use. It is a challenge to model both the imagery and behavior of a dynamic, interdependent and observable system. An important breakthrough in creating desktop VR and distributing it via the Internet was the development of Virtual Reality Modeling Language (VRML).

Several VRML applications have been demonstrated and they show promise in delivering three-dimensional content over the Internet using the unique capabilities offered by VRML. The types of VRML applications fall into several broad, but non-exclusive, categories. These categories include education and training, virtual collaboration, information visualization and retrieval, games and entertainment, design tools, and modeling and simulation. The application domains are also broad and include urban planning, geographic information systems (GIS), and medical, engineering, business and combat simulations.
Just as HTML became the standard authoring tool for creating cross-platform text for the Web, so VRML developed as the standard programming language for creating web-based VR. It was the first common cross-platform 3D interface that allowed creation of functional and interactive virtual worlds on a standard desktop computer. The current version, VRML 2.0, has become an international ISO/IEC standard under the name VRML97 [135] [136] capable of representing static and animated objects and it can have hyperlinks to other media such as sound, movies and images. Interaction and participation in VR web sites is typically done with a VRML plug-in for a web browser on a graphics monitor under mouse control. VRML is optimized for general 3D rendering and minimal network loading, taking advantage of extensive VRML browser capabilities. Visualization with VRML browser was proved to be an easy way of creating 3D and interactive content which may be modified and used without sophisticated and expensive modeling tools. Moreover, this content can be controlled by EAI interface.

Virtual Reality Modeling Language (VRML) is a scene description language which describes 3D environments over the web. VRML scenes are static in nature. The scene parameters needed for the visualization of the object and interaction with the world is provided in VRML. Visualizing a 3D program on a desktop needs a calculation of the 3D projection of the scene on the 2D display surface. It is applicable to the web environment also. The environments in a Web browser can be visualized through a plug-in called the VRML browser. Instead of sending a projection of the scene over the network, an instruction to create the 3D-scene is sent through the network and the user’s computer and VRML browser will do the actual work. Since VRML represent 3D scene graphs as an interchange format rather than a program, multiple small VRML models can be quickly composed into larger scenes. This approach permits construction of elaborate and consistent worlds from multiple independently created 3D worlds.

Prototypes allow encapsulation of objects and their geometry attributes and behaviors can be treated as parameters that receive values during runtime thus featuring dynamic binding. The PROTO node in VRML makes the objects created to be reusable. Prototypes are used to create new objects. The ROUTE statement in VRML helps to create dynamic scenes. ROUTES are a simple way of defining a path between an event generated by a node and a node receiving an event. Event-driven ROUTEs connect 3D nodes and fields to behavior-driven sensors and timing. Direct
interaction with the user is possible via sensors, interpolators and creation of intelligent animations using scripts. VRML97 allows the VRML browser to communicate with external applications, especially Java programs, giving the possibility of dynamic changes in a scene. The potentiality of VRML and VR browsers to manage dynamics and interact with the model is still underestimated. The primary interaction model for 3D VRML browsers is point and click, meaning that content can have embedded links just like the 2D Hyper Text Markup Language (HTML). This chapter describes how dynamism can be implemented in a .NET platform.

3.3 VRML concepts

VRML is a text based file format for describing interactive 3D objects and worlds. VRML is designed to be used either locally or over the Internet. VRML is also intended to be a universal interchange format for integrated 3D graphics and multimedia. VRML is capable of representing static and animated dynamic 3D and multimedia objects with hyperlinks to other media such as text, sounds, movies, and images. It is highly object-oriented with entities called objects and properties connected with the object, events and transformations changing the state of the object. VRML is also a hierarchical language. Primarily designed as an interchange file format, VRML models are structured as scene graphs in which grouping, transformation and selection nodes as interior branching nodes, while geometry, shape and accompanying render attributes are leaf nodes. A VRML file consists of the following major functional components: the header, the scene graph, the prototypes and event routing. A VRML file always starts with the header.

VRML scenes are capable of describing a wide range of 3D content. Scene graphs are used to hierarchically organize and manage the contents of the scene data. The objects and their behavior are represented at a high level. The scene graph establishes a clean boundary between scene representation and rendering. At the scene level, the concern is what's in the scene and any associated behavior or interaction among objects therein. The underlying implementation and rendering details are abstracted out of the scene graph. The VRML browser plug-in handles low-level concerns. The scene graph contains nodes that represent the objects in the scene. Nodes are connected by edges which establish the relationships among them. The nodes and the edges together produce a graph structure that organizes a collection of objects hierarchically. Every node except the entry node has a parent. Nodes that
contain other nodes are called parent nodes or grouping nodes. The contained nodes are called children or leaf nodes. The cumulative effect of nodes in a scene graph is called **state**. Their order is important as it creates a hierarchical tree, as shown in figure 3.1.

![Figure 3.1. VRML 2.0 Scene Graph Structure](image)

Geometric primitives such as IndexedFaceSet allow authoring tools and datasets to create highly complex objects. Textures and MovieTextures can wrap 2D images over and around arbitrary geometry. Sound nodes embed spatialized audio together with the associated shapes. Lighting and camera control provide complete control of presentation and rendering, including animation of camera position to create flyby explorations or dramatic visual transitions. Finally the Script node interface to Java and other programming languages allows modification or generation of any VRML content. The node is the fundamental building block of a VRML file which defines a part of the scene. Some nodes can be physical objects, such as a box, a cone, a spotlight, or 3D text; others are containers or provide special features. Containers can group nodes together to apply an effect on them or for later reuse. The VRML Standard defines semantics for 54 built-in nodes. They also support event generation mechanism. Nodes are used in the design of the scene to describe the geometry of the object both regular and irregular, illumination that provides directional, spot, point and ambient lights, materials and textures for mapping of JPEG, GIF, PNG image file formats. The VRML Standard defines nodes that can generally be categorized as:

- Geometry nodes that define the shape or form of an object.
- Geometric property nodes used to define certain aspects of geometry nodes.
• Appearance nodes that define geometry material and texture properties.
• Grouping nodes that define a coordinate space for children nodes they may contain.
• Light-source nodes that illuminate objects in the scene.
• Sensor nodes that react to environmental or user activity.
• Interpolator nodes that define a piecewise-linear function for animation purposes.
• Time-dependent nodes that activate and deactivate themselves at specified times.
• Bind able children nodes that are unique because only one of each type can be bound, or affect the user's experience, at any instant in time.

The Shape node is a container node which collects a pair of components called geometry and appearance which deals with geometry, colors and textures (Figure 3.2).

Figure 3.2 VRML shape node description

The appearance node takes care of the solid perception of the shapes, i.e. material (diffuse and specular colors) and texture. Texture to the surface of the objects is done by either wrapping a surface with an image or texture mapping. The texture mapping needs another node specifying the image coordinates corresponding to the coordinates of the geometry. The geometry of the 3D objects are described by using primitives
like cone, box, sphere, and cylinder or by sets of faces represented by coordinates of points. The node maintaining faces has two sections namely the coordinate section containing all the point's coordinates composing the object and the description section containing ordered lists of points constituting faces that serve as boundaries of the objects. The UML representation of a VRML object is shown in figure 3.3.

Figure 3.3 UML Notation of a VRML Object

Grouping is used to describe spatial and logical relationships between nodes, and also enable efficient rendering by 3D browsers. The Group node merely collects child nodes, with no implied ordering or relationship other than equivalent status in the scene graph. The Transform node similarly groups child nodes, but first applies rotation, scaling and translation to place child nodes in the proper coordinate frame of reference. The Billboard node keeps its child nodes aligned to always face the viewing camera, either directly or about an arbitrary rotation axis. The Collision node lets an author specify a bounding box which serves as a proxy to simplify collision detection calculations for grouped child nodes. The Switch node renders only one (or none) of its child nodes, and is useful for collecting alternate renderings of an object which might be triggered by external behaviors.

The LOD (level of detail) node also renders only one of multiple child nodes, but child selection is triggered automatically based either on viewer-to-object distance or on frame rate. Thus LOD enables the browser to efficiently select high-resolution or low-detail alternative renderings on-the-fly in order to support interactive rendering. The Inline node allows importing additional 3D data from another VRML
world into the current VRML world. In contrast, the Anchor node creates a link between its child nodes and an associated Uniform Resource Locator (URL) web address. When the child geometry of an Anchor node is clicked by the user’s mouse, the current VRML scene is entirely replaced by the VRML scene specified in the Anchor URL. Multiple strings can be used to specify any URL, permitting browsers to preferentially load local copies before searching for remote scenes or backup locations. Typically browsers highlight “hot links” in a 3D scene by modifying the mouse cursor when it is placed over Anchor-enabled shapes.

Virtual lights in a 3D scene are used to determine illumination values when rendering. They are used to calculate visibility, shininess and reflection in accordance with a carefully specified mathematical lighting model. The DirectionalLight node illuminates using parallel rays, the PointLight node models rays radiating omnidirectionally from a point, and the SpotLight node similarly provides radial rays constrained within a conical angle. Lights include color and intensity values which are multiplied against material values to calculate produce proper shading. The Sound node places an audio clip at a certain location, and with non-rendered ellipsoids surrounding it determines minimum and maximum threshold distances. Sound can repeat, be rendered spatially relative to user location, and be triggered on/off by events. Embedding various lights and sounds at different locations within a scene can produce dramatic results. Figure 3.4 shows the various nodes that are grouped to construct an object.

![UML Representation of Interpolator, Sensor and Light nodes](image)

Figure 3.4 UML Representation of Interpolator, Sensor and Light nodes
Sensors detect change in the scene due to passage of time (TimeSensor), user intervention or other activity such as viewer proximity (VisibilitySensor). Sensors produce time-stamped events whose values can be routed as inputs to other nodes in the scene. User intervention is often as simple as direct mouse interaction with a shape via a TouchSensor, or interaction with a constraining bounding geometry specified by PlaneSensor or SphereSensor. ProximitySensor checks if the user is near enough the element in the 3D space. When the predefined sensors are not sufficient EAI can be used to obtain complete monitoring of the user behavior.

Consistently typed input and output events are connected to correspondingly typed fields in the scene graph via ROUTEs. Key-frame animation typically consists of simple time-varying values applied to the fields of the appropriate node. Smooth in-between animation can be interpolated linearly as long as key values are at sufficient resolution. Linear interpolators are provided for Color, Coordinate, Normal, Orientation, Position and scalar fields. Linear interpolation is sufficient for many demanding applications.

Prototypes (PROTO and EXTERNPROTO) allow creation of new VRML node types by authoring combinations of nodes and fields from other preexisting node types. In this sense, a PROTO definition is somewhat analogous to a macro definition. In order to avoid completely copying a PROTO for each file where it is used, the EXTERNPROTO definition specifies remote URL where the original PROTO is defined, along with the interface to permit local type-checking by browsers during scene loading. The EXTERNPROTO mechanism thus allows construction of PROTO libraries for easy access and reuse. Prototype definitions can be included in the file in which they are used or defined externally. New node types in terms of already defined “built-in” node types or user-defined prototypes can be defined with prototypes. All node type names should be unique in each VRML file. A PROTO statement is used to declare a prototype interface which can define the event interface as well as default values for prototype’s fields. Each prototype instance can be considered to be a complete copy of the prototype, with its own fields, events and copy of the prototype definition.

VRML supports a Script node for complicated movements and manipulations that facilitates dynamic behaviors written in programming languages such as ECMAScript, JavaScript, and Java. Interfaces between them are done through Script nodes, an event engine, DEF/USE naming conventions, and ROUTEs connecting
various nodes and fields in the VRML scene. Script nodes encapsulate the program functionality, and ROUTEs provide the wiring that connects computation to rendering. They provide flexibility to expand the behavior requirements of an object. They are typically used to signify a change in the scene or some form of user action, receive events from other nodes, and encapsulate program modules that perform computations or effect change elsewhere in the scene by sending events. This is a way to create the dynamically changing VRML-model. Within a script one can dynamically direct events for any node existing in the model. Script nodes allow user-defined event processing. Script node is used to program behavior in a scene. In summary, Script nodes typically

a) Signify a change or user action.

b) Receive events from other nodes.

c) Contain a program module that performs some computation.

d) Effect change somewhere else in the scene by sending events.

Events allow VRML scenes to be dynamic. When a user interacts with an object, an event is sent to the server requesting details on the available information associated to the object. These events are processed by the server and the resulting information, adapted to the current session parameters, is sent back to the client. Events are merely time-stamped values passed to and from different parts of a VRML world. EventIns accept events, and EventOuts send events (when triggered by some predefined behavior). Events must strictly match the simple type (such as integer, float, color) or node type (such as a Material node) being passed from input to output. Script parameters are designated as eventIn, eventOut or exposedField, which respectively correspond to in, out or in/out parameter semantics. Private Fields are simply designated as field rather than exposedField.

Figure 3.5 Processes of VRML Events
The event routing mechanism allows the connection between nodes generating an event and node receiving an event. When a node receives an event, it reacts accordingly by changing its state, which might trigger additional events. Nodes can change the state of objects in the scene by sending events. A node's implementation defines how it reacts to events, when it may generate and send events, and any visual or auditory appearance it might have in the scene.

Event routing separates event from the scene graph hierarchy. Every time an exposed field is changed an event is generated. Once generated, events are sent to their routed destinations in time order and processed by the receiving node. When an event occurs, the node which generated the event outputs a value or set of values of a given data type depending on the type of event. When one sends an event to a node, one is sending a value or set of values to that node. The node determines what the event should do with the value or values provided. In the VRML nodes, when a node receives an event it alters one field, the field specified by the event. This processing can change the state of the node, generate additional events, or change the structure of the scene graph. Figure 3.6 shows the execution model of a VRML browser.

![Figure 3.6 Execution Model of a VRML Browser](image)

Node naming and light-weight multiple instancing of nodes is possible through the DEF (define) and USE mechanisms. DEF is used to associate names with nodes. USE permits duplicate instances of nodes to be efficiently referenced without complete re-instantiation, significantly boosting performance. Node names created via DEF are also used for routing events to and from fields. Thus Script nodes (and other 3D nodes which interact with the script) all must be named using DEF. The scope of all DEF’ed names is kept local to the file (or PROTO) where the name is defined.
ROUTE statements define connections between named nodes and fields, allowing events to pass from source to target. ROUTE statements usually appear at the end of a file since all nodes must be DEF’ed (named) prior to referencing. Typically ROUTEs are used for all events passed into (or out of) Script nodes. Use of ROUTEs is not always required, however, since nodes in the VRML scene can be passed by reference as fields to the encapsulated Script code. This second approach permits direct manipulation of VRML by a programming language without using events or ROUTEs.

Nodes like sensors, routes and interpolators are combined to introduce dynamism in the 3D worlds. User position and actions like mouse move, click and drag are detected by sensors. They also identify the change in time. The events occurred are captured and directed to interpolators to change the values of certain fields like color, position, orientation and scale by routes. Thus the animation provided in VRML is often insufficient to describe complex actions. Real-time navigation of a 3D world on the web is possible by a VR browser. The dynamics is provided with a sensor which has to be described in advance. The VRML world can be modularized into smaller modules or worlds and can be assembled together. The Inline node is used for such organization of large data. The UML representation of a complete VRML object comprising of all the nodes available in VRML is shown in figure 3.7.

Figure 3.7 UML Notation of Complete VRML Nodes
3.4 OOPs and VRML

The concept of object-oriented technique is practiced during the process of real-time data access and scene simulation. OOP (Object-Oriented Programming) is a method of packaging data and codes. Class, encapsulation and inheritance techniques are used in the structure that can make full use of the benefits of object orientation, such as hierarchy and polymorphism. In traditional systems, the servers do the work of business logic, and the clients show messages to users. Dynamic E-Learning through Virtual Reality is based on these concepts and Web3D process the data in the server, and converts results into highly efficient serialized messages before transmitted to the client. After that, the client transforms messages into attributes of objects defined by JavaScript, which can help to operate entities in the virtual scenes and interact with users. The concept of object orientation is realized in both the server and the client, particularly in the 3D interaction.

3.5 Drawbacks of the VRML language

It has advantages such as small file size, real-time shades, full range of viewing, VRML becoming a new generation of 3D network technology has been widely used. But most of the VRML works still remain in the static display stage, lack of dynamic and interactive, mainly in the following points [137].

- VRML specification does not maintain the semantic descriptions of models. Hence other standards like HTML are used.
- The VRML hardly communicate with the HTML file, just as plug-ins embedded in Web pages [138]. The VRML data can not transmit to HTML. HTML also cannot read VRML data.
- VRML interaction is only limited to a mouse click or move and can do nothing for the requirement of input data from user. Since there is no nodes to display decision logic when behavior changes, the ability to interact with the outside of VRML is not very strong. To create a complex 3D interactive scene, it must be combined with other language to construct virtual scene which has interactive ability [139].
- Although interaction is simulated through sensors, interpolators and routes, no editing tools or modelers are available to manipulate the objects.
• The design of a VRML document capable of introducing and controlling dynamics on the basis of static information is impossible task for CGI scripting.

• The browsers do not save the VRML world.

• In many cases, VRML applications had to be extended to include things VRML lacked, such as sophisticated user interface and interactivity, database access, multiuser support, security and system integration support.

• Applications based on the EAI would not work reliably due to changes in the client environment by competitive commercial stakeholders, affecting things like the support for a required third party programming language.

3.6 Integrating VRML with Web Services

The development of Web 3D technologies help in realizing realistic scene situation with intelligent and dynamic interaction in web content. The two dimension simulation without visualization and reality cannot describe the layout and status of the realistic scenes, while the videos cannot express the complicated relations between events and the scope, angle and video clarity are easy to be effected by the environment. The transmitted data is too big on the network. In such cases VR is an effective technology to construct a platform with high performance [140]. The VR technology makes use of computer graphics technique, sensor technique, human-computer interaction technique, network technique and others. The publication of Web3D standard made it possible to implement VR on the Internet. The Web3D technology that includes related languages, protocols and software development tools, is a new developing method to construct 3D virtual reality [138-140]. Web3D is similar to other web techniques using the same concepts - the content is stored in the web server, the clients send requests to get the response data that tend to be displayed in browsers. To heighten the efficiency, extensibility and platform independence of system, many functions are designed to retrieve and assemble when needed.

The clients’ browsers display the virtual world, interact with users directly as well as update entities in the world according to messages received from the Web server. Therefore, the most important part of system is the VRML interface, which has essential effect on the scene information and interaction experience with users. Actually, VRML interface has to add, delete or modify the status of entities, listen to the events outputted from the whole scene and respond to them through the way of
routes and functions. In virtual scenes, entities contain visible objects and also invisible like sensors. All of the entities are instances of VR classes dynamically defined by the system. They have their unique identities, attributes and functions that are designed to meet the need of input or output events. These methods guarantee that initialization and update of virtual scenes, dynamic consistency of entities and the web server data are well performed. The requirements for such a system are the following:

- **Service access through a web interface.** Using the web as main gate to the services confers an easy, world-wide accessibility and a direct exploitation of the WWW facilities on the system.

- **Browsing featured by a dynamic virtual world.** A dynamic virtual world resulting from the integration of a VRML world and C# enhances the plain navigability features of hypertexts and improves the user experience compared to a traditional “click-based” GUI.

- **Highly-interactive service.** The whole scene is rendered immediately based on the Internet standard media streaming and control techniques and protocols.

- **Multi-platform portability.** By relying on the .NET platform, Web technologies, VRML and the Internet multimedia protocol stack [141], the system is endowed with an inherent multi-platform portability.

While generating a virtual scene, the feel of realistic experience depends on the rendering of visible entities. A better scene is achieved by improving the model details like the clarity of texture and images, audio and video in the background. The real-time performance of the system can be achieved by following certain rules in the modeling process.

1. Basic shapes are preferable, complex models are developed by Boolean operation, and redundant vertices and edges are removed without reducing the model details to cut down the number of facets and the volume of file.

2. Compared with meticulous models, models that adopt texture images can not only render vivid scenes, but also reduce the complexity of modeling. The JPEG images have good features of avoiding distortion and outstanding graphic effect.
3. Models are unique named and divided into groups to optimize the scene hierarchy. These rules help to edit the scripts and functions to respond to events timely. On the other hand, the system focuses on the real-time monitoring, which includes implement of dynamic entities and interaction [142]. In order to manipulate entities in an object-oriented fashion, one method that is similar to the concept of class of OOP is designed to encapsulate attributes, input/output events, routes and predefined functions into different entities. The method can save bandwidth resources, define prototype structures and instantiate entities when needed. Every instance is a duplication of predefined prototype structures and has its unique attribute values and underlying events. To manipulate entities dynamically, a PROTO nodes library is needed.

The use of a proto node generates a new node as defined with a field id which is its identity in the virtual world. Interactive operations and input events will lead to the state change of node, thus affecting the entities in scenes. The enabled field changes availability of entities, the transparency field changes visibility and the rotation field adjusts entities to get a right position in scenes, etc. Nodes affected by a given operation are visited; its internal state may be set or altered so that it reflects the state of the operation at that point in time. Rendering traversals occur almost constantly with interactive and animated graphics because the state of affairs changes as often as the user's viewpoint, necessitating continual scene graph queries and updates in response to an ever-changing perspective. To get a more realistic scene simulation, the work of designing entities’ fields, events and functions at length is necessary.

The parameters received from external programs must be mapped into corresponding nodes’ fields, so the first step is to get the right node in line with the id. As the data from the database or sensors is upgraded in the web server, or the scene in the client generates events, the server transforms the result into serialized message in the format defined in the messages protocol. In receiving messages, the JavaScript functions and VRML interface functions work together to update the virtual scenes in time. Object oriented programming is realized in the script program by this way, which contributes to interacting with entities of virtual scenes. JavaScript can be used for generating quick code that need not be complex. It handles events very comfortably. The script has the ability to communicate with the browser. It provides the ability to dynamically load or create new objects and ROUTEs on the fly. For
example, when an object is clicked in the scene, the script behind the click may create a new object and place it in the scene dynamically.

VRML applications have evolved, first through development of the client part [143, 144] and then have been extended to include client server applications [145, 146]. The client applications are stand-alone applications that support live updates, which is the ability to add features to the world without reload. To expand the capabilities, server processing has been developed to increase the flexibility and capabilities [144, 145]. The server processing provides the opportunity to enforce access control, selection of the content to be viewed, and generation of content dynamically. A key challenge in creating visualizations is being able to provide timely, up to date information to clients automatically. In order to achieve this flexibility, the server needed to be able to determine what information was available on the server and then present the appropriate choices to the user.

3.6 System Architecture for Dynamic Generation of 3D Web Content

The design of dynamic generation of 3D web content enables server side user interaction, dynamic composition of virtual scenes, access to on-line data, and selection of content, personalization and implementation of persistency. The VRML document is created on the fly, according to the input given by the user in an HTML form. The input is mainly the properties for building 3D scenes. The given input is processed by a CGI script which creates the VRML document. It is sent to the VRML-enabled www browser which renders the world for display and navigation. In this approach, the VRML scene models are generated by the application instead of being retrieved from previously prepared files. The architecture used for generating VRML scene on the fly is presented in figure 3.8.

![Figure 3.8 A Simple architecture for dynamic generation of 3D web content](image)

The interpretation, execution, and presentation of VRML files will typically be undertaken by a mechanism known as a VRML plug-in browser, which displays the shapes and sounds in the scene graph. This presentation is known as a virtual world
and is navigated in the browser by a *user*. The world is displayed as if experienced from a particular location; that position and orientation in the world is known as the *viewer*. The browser may define navigation paradigms such as walking or flying that enables the user to move the viewer through the virtual world. In addition to navigation, the browser may provide a mechanism allowing the user to interact with the world through sensor nodes in the scene graph hierarchy. Sensors respond to user interaction with geometric objects in the world, the movement of the user through the world, or the passage of time. During a session, the user has the ability to make live updates, i.e. modify the world by changing their parameters.

The generated virtual scenes reflect the different user requirements regarding their contents and form. The browser reacts on user actions (other than navigation) only if they are initially and explicitly described in the VRML document. A particular sensor has to be attached to a particular object before the user is able to interact with that object. The next step is the composition of the response. In this approach, the decision on the type of sensor, the target object and the resulting event (CGI script or Javascript, or appropriate VRML nodes, or files on remote servers), has to be taken by the CGI script during the dynamic creation of the document. An extended version of the architecture and its components is shown in figure 3.9.

![Figure 3.9 Architecture for dynamic generation of VRML scenes with its components](image-url)
Dynamic objects refer to the objects that have been generated using 3D coordinates based on the user input in the HTML form. The real time simulation for the dynamic objects is implemented using Script node. An external script programme EAI (C# file) reads the real time value especially the size of work piece in the control panel and transfers the value to the Script node in VRML. Figure 3.10 shows a common architecture of VRML browser [147].

![Figure 3.10 Architecture of VRML browser](image)

3.7 Implementation of Dynamic 3D Web Content Generation

The application Dynamic E-Learning through Virtual Reality (VREL) is developed. VREL features the study of three dimensional concepts using virtual reality. Basic primitive 3D shapes are generated on the fly upon users input in an HTML form. The user can interact with the generated primitive 3D shape and change its parameters (color in RGB format) to view the possible changes in the 3D model. The X-Y axes are drawn and the axis of rotation is also shown for easy comprehension. Slider pots are provided to change the values of the parameters. IIS and a dot net platform were used for implementation. PROTOs were used for the extensive interaction used in this application. Table 3.1 lists the PROTOs used with its functionalities, parameters if any, eventIns and eventOuts.
Table 3.1 Prototypes used in VREL

The steps involved in the design of VREL are as follows.

1. The homepage of the website is designed with frames as shown. The top horizontal frame displays the application title. The next frame consists of the subject topic and the menu to be used for study. The last frame is divided into two columns indicating the content of display in them. Figure 3.11 shows the home page of the application.
2. An option from the menu (a shape) should be chosen. An input form is displayed with the needed parameters for the construction of the 3D shape. The input webpage is shown in figure 3.12. The input values of the parameters should be specified as required and the MakeVRML button must be clicked.
3. The parameters values are passed to a CGI script program (C#) which encodes the VRML coding for the given 3D shape with its values. The slider scroll is designed as a prototype. The C# program uses EXTERN PROTO for the sliders. The encoded VRML coding is saved as a wrl file and directed to output in the specified frame. The description of the generated 3D model is displayed to the left of the VRML scene. To enhance user interaction, sliders are designed to change the RGB colors. The object that is displayed on the browser has the XYZ orientation as (0 1 0).

The UML representation of rendering a cube is shown in figure 3.13

![UML representation of rendering a cube](image)

**Figure 3.13 UML representation of rendering a cube**

The object that is on the display can be zoomed, orientation changed, moved and scaled using the VRML browser plug-in (Cosmo Player). The three dimensional concept can be understood easily with the designed scroll sliders for x, y and z directions. The x-y axes are shown in the display. The z axis is perpendicular to the plane. The axis of rotation is also shown which is green in color. A slider to change the angle of rotation is also provided. The generated output for the given input parameters of a cube are shown in figure 3.14.

The principle lying behind this is as follows: Initially, a VRML file is downloaded to the local browser who renders its contents. The script nodes in the VRML scene are local Java scripts. Scripts are able to manipulate scene graph nodes by generating events that are delivered to the node and change one or more of its properties; for example, its position in the scene, its shape, or one of its material attributes. Obviously, since the scripts are fully functional Java codes, they are not
restricted to just changing the scene graph. They can, for example, dynamically generate additional VRML nodes, or locate and add existing VRML to the base scene downloaded in the original VRML file.

Figure 3.14 Dynamic generation of Cube

4. The color mixer slider can be varied and the changes in the color can be noted. Similarly the orientation of the model can be varied by changing the axes slider. The angle of rotation can also be varied and the dynamic change in the model can be noticed. The modified model after the application of variations using the scroll sliders is shown in figure 3.15.

The important highlight at this point is the fact that from the user’s point of view, the modification processes are completely transparent as there is no need to reload the 3D graphics into the Browser each time the session parameters are modified. Other 3D geometric shapes like box, sphere, cylinder and cone can be generated in similar fashion. The input parameters of the different 3D primitives differ and the corresponding input webpage will be displayed when selected. The modifications can be made dynamically as by the choice of the user. The parameters for box are height, width, depth and the RGB colors. The parameters for Sphere are radius and the RGB colors. The parameters for Cylinder are height, radius and the RGB colors. The parameters for Cone are bottom radius, height and the RGB colors.
Transformations like Scaling and Translation can also be implemented for the 3D objects in a similar way. Sliders can be introduced for varying the parameters of a VRML node like position, viewpoint, DirectionalLight and Material node. This application helps a beginner to master the 3D concepts easily. The rotation about an axis is clearly understood which is difficult to comprehend in three-dimension.

VR based educational model gives the student a lot of potential. The dynamic association of information had been achieved improving the learning process for the students as it had been adapted to their particular needs. In conclusion, the system implementation has demonstrated that the strategy proposed is applicable within an e-learning platform. Building complex shapes is also possible in this design by adding the primitive nodes.