The Nottingham Trent University.

Computing Department.

The Pentacle Method: Structured VRML Development.

Drew Baker.

1997.

Project Report in part fulfillment of the requirements for the

degree of Bachelor of Science with Honours in

Computer Studies.

Abstract.

This paper examines the application of current software engineering paradigms to the new Virtual Reality Modelling Language (VRML), a specialised three dimensional graphics language, developed as a companion to the World Wide Web Hyper Text Mark-up Language (HTML).

The paper initially presents two hypotheses; 1) Current software engineering paradigms are not well suited to VRML development, and, 2) If VRML is to be accepted as a quality medium by both the computing and business community then such a paradigm must exist.

In order to establish the validity of these two hypotheses the paper details research into the current methods used by VRML developers, and compares these with three established software engineering paradigms. From this research the principles of the methodology movement are discussed and their application to the requirements of the new VRML media considered. Using these themes, a paradigm supporting the processes used in the creation of VRML systems is developed. The derived method is based around a simple development model that attempts to cater for the project management, quality assurance, software engineering and the graphic art requirements that must be considered in a project using VRML.

The method as presented is applied to a world building project scenario and illustrated by the development of a component world. The applicability of the derived method is considered in the light of the scenario VRML development, and the evaluation critically discussed.

The paper concludes that the presented hypotheses are valid, and that the derived methodology, while not perfect, represents a significant improvement to the current support paradigms for the VRML development process.

Acknowledgements

The author would like to express his sincere gratitude to the powers within the universe, in *whatever* forms they make themselves manifest, for granting him life, liberty and the gift of intelligence, without their benefice this work would be left incomplete.

Thanks are also extended to Paul McDonald, the project supervisor, who took a vague idea of the project domain and laid the foundation for the work the reader now holds.

The patience and support of the author's partner Zoë Tew is also gratefully acknowledged, for without her acquiescence, the time and resources available to the production of this work would have been severely limited.

The author would also like to express his deep appreciation to his housemates, Jane Breach, John Tribe and Paul Wilson. These friends have put up with both the evangelical fervour and wails of despair from a crazed person absorbed in his work and provided the author with welcome coffee at unexpected and often unsociable times of the day.

Finally the author wishes to acknowledge the late Malcolm Hill, a friend, a confidant fatally blessed with genius, to whom this work is dedicated. Good night sweet prince, and flights of angels guide thee to thy rest.

"The last hope before the next incarnation."

Contents.

Contents *Page*

List of Figures.

1 Introduction

"There are those who think that Virtual Reality may be the most important development since man first chipped flint, and there are those who don't know what it is yet"

Douglas Adams (1991).

In the 1984 novel "Neuromancer", the science fiction author William Gibson described a near future world where humans live, work and play within the "matrix", a vast network of linked computers. The matrix forms the backbone of, what Gibson termed *cyberspace*, a virtual space composed of digitally created artificial realities.

Gibson's vision of cyberspace relies on the synthesis of two distinct factors to achieve its "consensual hallucination", communications and graphical representation of the cyberspace environment, objects and inhabitants to create the illusion of reality through the distributed systems that comprise the matrix.

However much Gibson's futuristic world seemed fantastic to the science fiction readers of the mid 80's (and even to the present day reader), Gibson predicted the advances in communications and technology that have lead to the realisation of the Internet as a communication medium for the common computer user.

Cyberspace is, of course, a work of fiction (although may aspects of computing and society have latched on to the phrase), however, Gibson's vision has provided the impetus for serious research to be conducted into developing interactive and distributed digital environments for both business and pleasure.

1.1 The Origins of VRML.

The development of the Internet in its many different guises is a matter of historic record but its phenomenal rise in popularity over the past decade can be primarily, and arguably, attributed to one single factor, the development of the *Hyper Text Mark-up Language* (HTML) protocol. Developed by Berners-Lee (1990) as part of an initiative of the European Laboratory for Particle Physics in Switzerland (CERN) in 1989. HTML is a descriptive language enabling the creation of "pages" for the Internet's "World Wide Web" (WWW).

The HTML protocol provides a platform independent system for writing and displaying text and graphical images accessible via an HTML aware browser. The protocol provides methods for linking pages together to form a "web" of pages connected by a common thread (subject matter, common interest, author's preferences, etc.). Although HTML is a user friendly medium for disseminating information, a HTML page is essentially a two dimensional and static medium for the presentation of information to the web user (although subsequent revisions of HTML have added support for two dimensional animation, sound and a degree of interaction).

The discipline of cognitive science defines the human thinking and data retention processes as one of objects, their behaviours, attributes, and interrelation with other objects. While two dimensional mediums such as text and graphical images provide a common interface to the majority of people for understanding and acquiring information such devices are an abstraction of reality of the subject in hand. The problem of communicating ideas and concepts that are intrinsically three dimensional in nature via two dimensional mediums has been addressed in many different ways by many different ages, from Giotto's formalisation of perspective in the 15th century to modern day engineering and architectural blueprints.

Modern two-dimensional media such as film and television have addressed the issues of static information dispersal as the "experience" but have still not overcome the obstacles of interaction with the subject matter. The dynamic interaction with data and information has

arguably not been available to mankind until the advent of the computer. Indeed the accepted roots of modern electronic computing, the ENIAC project (and to some extent Babbage's mechanical difference engine), lie with the need to simulate artillery telemetry from constantly revised battlefield data.

The possibilities of using the computer to provide a digital artifice where the user could interactively control a simulated experience was first presented by Sutherland in 1965 in his seminal paper "The Ultimate Display". Sutherland's paper detailed his research into a head mounted output device specifically for the use in interactive artificial environments. "The Ultimate Display", and subsequent research and development in the fields of three-dimensional interactive graphics and their uses, earned Sutherland the title of the "father of virtual reality". Although it is noted that it was not until 1983 when Lanier actually first suggested coined the terms virtual reality and its common abbreviation VR

Further research and development of the virtual reality medium was conducted during the 1970's, mainly by organisations interested in training in and visualising environments either that did not exist or where there was a large inherent financial risk. Most notably these "pioneers" of virtual reality composed of the military, avionics, large scale architectural projects and the scientific community. These early virtual reality systems tended to be bespoke in their nature and very expensive due to the required computational power and supporting technology required in the realisation of the interactive environment.

The technological advances of the early 1980's and the fall in price of computer hardware soon gave birth to an increasing portfolio of companies offering virtual reality commercially. Initially restricted to the "traditional" uses of VR, the application of virtual reality technology soon became applied to such diverse commercial areas such as medicine, cartography, entertainment and manufacturing.

Virtual reality's diverse and adaptable usage was reviewed in the paper "Practical Applications of Artificial Reality" co-written by the author in 1994. The paper concluded that the wide spread commercial usage of virtual reality did not exist, it was further concluded that the reasons behind this non-acceptance of the medium were as follows:

1. The medium was "to new" and "unproven" for business to fully realise potential application areas.
2. There was no *common* language to create virtual environments with (such as basic, c, or HTML).
3. The software development environments for virtual reality that did exist were based around individual platforms and required vendor specific software to interpret, display and interact with the created objects and environments.

The paper considered that until these issues could be addressed the full potential of virtual reality would not be realised.

While the paper was being written a discussion regarding the development of a common language for describing three dimensional objects and environments was being hosted by the Internet related magazine "WIRED" (1994). In October 1994, after much debate, a proposal based on Silicon Graphic "Open Inventor" 3-D metafile format was selected and published as the *Virtual Reality Modelling Language* (VRML) version 1.0 specification (Bell, Behlendof and Pesce 1994).

The specification was well received by the computing community and the VRML Architecture Group (VAG) was formed to moderate and co-ordinate the development of the new language. The VAG itself has representation from ten 3-D computer graphics leaders most notably IBM, Silicon Graphics and Sun Microsystems.

VRML is seen by its exponents as having the potential to redefine the graphical representation of, and access to information in 3-D on the WWW in much the same way which 2-D HTML has impacted access to information on the WWW over the past decade (Rose (1994)). Like HTML, VRML provides an interpreted, platform independent, language that can be viewed by any VRML compliant browser not specifically tied to one specific vendor. While the use of a VRML browser does not provide the immersive experience usually associated with virtual reality it provides a desktop view of the virtual environment, commonly called “window on the world”.

If the conclusions of the 1994 collaborative paper are correct, then, VRML has the capability for bringing virtual reality into the main stream market as a data visualisation and information access tool.

1.2 The Development of Cyberspace Using VRML.

To date the VRML specification has undergone one minor revision (for clarification of some vagaries in the original specification) and a major revision in June 1996. The VRML version 2 specification includes the provision integration of the Java programming language into VRML files, this inclusion has considerably enhanced the language from a language describing static 3-D objects to a dynamic and interactive 3-D environment.

The comparative newness of the language suggests the apparent lack of quality textbooks that are available on the subject of VRML (for either of the two versions). Indeed even the most casual review of the so-called VRML publications reveals that many of are in fact either narratives on the Internet, virtual reality or general 3-D graphics.

Online research material via the Internet provides those who are interested in constructing VRML environments (referred to as worlds) and their component objects with a wealth of examples, tutorials and technical notes on the subject. It is observed, however, that there is

distinct lack of any *structured* approach to the construction of VRML worlds. Indeed, it is noted that the majority of texts on the subject offer a “code and fix” (Manns and Coleman (1996)) style of approach to the development process.

Like HTML, the construction of VRML worlds is accessible to anyone who understands the format of the language and has a text editor to create the VRML script. While this accessibility to the medium is considered as being critical to the success of the medium, it may prove to be its ultimate downfall. It is observed that the rise in popularity of HTML has increased the amount of data and traffic on the WWW. However, because the majority of HTML producers utilise the code and fix approach many of these pages have not employed any design principles or considered human computer interaction factors in their construction. Such undisciplined approaches to HTML page construction can lead to the page reader not understanding the content, becoming lost within the web, becoming frustrated or waiting an inexorable age for the page to be downloaded.

1.3 Introduction Conclusion.

If VRML is really to be the next generation of WWW access and information medium then it is considered that quality *must* be of prime concern to the pioneer world builders. While many software design methodologies and paradigms exist to support traditional software engineering their distinct absence from current dissertations on world development suggests that either

1. The use of such paradigms by world builders is assumed (although this would preclude the necessity for the much offered code and fix approach)
2. Or, that the existing methods do not well support VRML and the essentially graphical nature of the medium.

The 1994 collaborative paper identified the primary factor of the slow uptake of virtual reality per se as the unwillingness of business to invest in new and unproved technology. If this conclusion is correct then the adoption of the "Heath Robinson" approach to VRML world development will not endear the medium further to the commercial customer. It is therefore considered that if VRML is to be taken as a serious and quality medium for the representation of data and dissemination of information across the WWW then a paradigm for world building, or rather world engineering must exist.

This report is, therefore based on two simple hypotheses:

1. The design requirements of VRML objects and their worlds are not well suited to the existing software engineering paradigms.
2. If VRML is to be a quality communication medium such a paradigm must exist.

The report will examine the existing methods of design and assess their limitations concerning the construction of VRML worlds (chapter 2). This research will provide the basis of a support

paradigm for the design needs of the new language (chapter 3). As such the paradigm will be required to address the following world building aspects of VRML;

1. The construction of worlds and their objects.
2. The manipulation of objects within the world.
3. The planning of object behaviour.
4. The presentation of VRML objects and worlds.

In order to provide holistic support for VRML world design, the methodology will also consider the following aspects;

1. The role of the world user.
2. The role of the world browser.
3. The specification of worlds.
4. The design of the human computer interface to the world.

The derived paradigm will be presented (chapter 4) and used to develop a function VRML world (chapter 5). The process of developing the world will be assessed to establish the value of the method as a future support tool for quality VRML world design (chapter 6).

2 Limitations.

"When Columbus was sailing west in his quest for a round earth, he reached a point where he was hopelessly lost. Legend has it that he gathered all the sailors and exclaimed 'We have arrived at uncharted waters much sooner than I had anticipated. Rejoice!'. This is the current situation with the web."

John R. Vacca (1996).

The initial research for the project had consisted of reviewing texts, journals and papers on the subjects of 3D graphics, computer data representation and virtual reality in order to determine if any existing paradigms existed for VRML aligned disciplines. The need to research the VRML aligned disciplines, as opposed to VRML itself, became apparent due to the lack of available documentation on VRML world design itself (this it is suggested is attributed to the relevant "newness" of the language). The research indicated that none of the current software engineering paradigms were being specifically applied to the problem domain of the literary subject (although it is conceded that such paradigms may well have been utilised in whole or part but their use been unaccredited).

In order to establish the presented hypothesis as facts the need to research and review any already existing paradigms for VRML world building, and the extent to which they were being used by the world building community, was clear. Subsequent to the investigation of possible VRML world design support methodologies the hypothesis that the popular software engineering paradigms do not cater well to the needs of world building also requires substantiation.

It was considered that the investigation of the hypothesis would need to be conducted on two fronts, at a high level reviewing new VRML related articles and at a low level by contacting world builders themselves to determine their individual approaches to world development. In order to provide a conduit for world builders to express their views and usage of design methods

for VRML, postings were made to various VRML\VR news groups on the Internet and a number of businesses expressing commercial interest in VRML construction were contacted to solicit their opinions.

2.1 Existing VRML Development Approaches

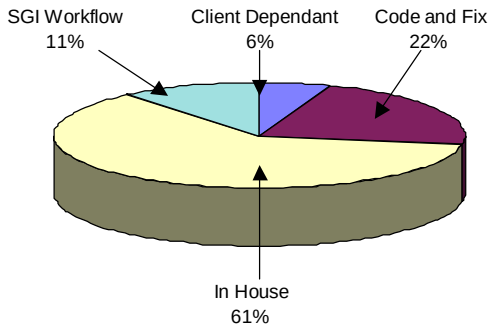
In order to establish the whether or not world builders were utilising any forms of design paradigms for VRML world and object creation postings were made to the four existing news groups specifically dealing with VRML discussions. Although these news groups had the potential to solicit replies from a wide range of interested parties from the VRML community the response to the posting was poor with a total of only eighteen replies (8 commercial, 10 independent).

The lack of response by the news group participants and content of the news group topics, suggest the following general observations:

1. World builders do not widely use specific design models in VRML construction (assumption).
2. Commercial world builders are unwilling to discuss in-house development paradigms.
3. The technical aspects of VRML are currently being explored and as such the methods of developing quality VRML worlds and objects are not being considered as a high priority by the VRML community at large.

The survey identified a number of VRML development approaches currently being utilised within the world building community. The results of the survey are presented in figure 1 opposite.

Figure 1: Analysis of Survey Response.



It is considered that a brief review of these approaches is necessary to support the original hypothesis.

2.1.1 Client Dependant Methodologies.

Two of the commercial respondents indicated that the approach to the development of VRML applications depended on the restrictions imposed by the commissioning client.

Although this practice was considered to be the exception to the rule by the respondent's reference was made to a number of contract VRML worlds that had been developed as part of larger software development projects. The addition of the VRML components had to comply with the software quality plans for the host development project (notably the use of the ISO 9000 and ANSI/IEEE 730-1984/983-1996 standards).

Both of the respondents observed that the “tweaking” of existing software engineering approaches used on such projects was not an optimal solution to developing VRML worlds and that whenever possibly an attempt to persuade the client to consider in house methodologies was made. Although the respondents could not comment on either the software engineering paradigms used or problems encountered with such on the projects it was noted that there is a significant trend towards the requirement by clients to conform to a recognised standard for world building.

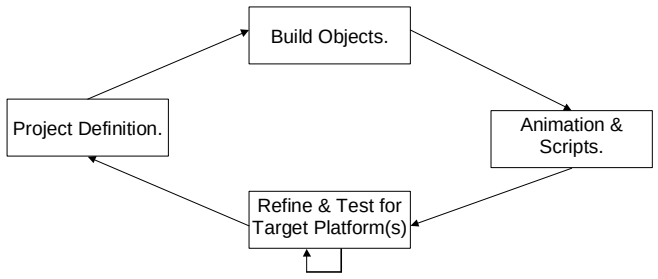
2.1.2 SGI Workflow.

Silicon Graphics Incorporated, one of the "pioneer" companies of VRML, has produced a document to aid the world builder in the process of developing VRML applications. The document, "Project Workflow" (SGI (1996)), is not large (fitting on one page of A4) but attempts to encapsulate the considerations that the world builder must make before the commencement of a VRML project.

The SGI Workflow approach is therefore not a complete design paradigm, but rather is a profile of the major milestones that should be reached during a VRML development project.

The Workflow approach is presented as a cyclic development process of four parts each paying attention to different aspects of the development process (using the Workflow outline this process is graphically represented in figure 2 opposite).

Figure 2: Representation of the SGI Workflow Approach.



1. Project Definition.

Workflow advocates the use of story boarding techniques (Cox and Walker (1993)) to visualise the interaction between the world participant and world components for visualising the attributes and content of the world. The project definition also requires that the developer have a clear understanding of both the participant uses of the world and the platform that the participant will be browsing the world from (see 2.3).

2. Build Objects.

The Workflow process acknowledges that not all of the world objects will need to be created from scratch but may be reused from other projects, purchased or converted to VRML format from other 3D file formats (such as CAD, or geometry modelling applications).

3. Animation and Scripts.

The addition of animation and behavioural scripts into the VRML world are methods by which the participant can interact with the objects of the virtual environment. The VRML version 1 specification provides the world builder with the ability to create static 3D representations with limited interaction (often referred to as *dead* worlds). The refinements included in VRML version 2, however, include provision for logic and behaviour to be attributed to the world components by the Java language (hence the common reference to version 2 as *moving* worlds).

4. Refine and Test for Target Platform(s).

Once the world has been constructed the Workflow suggests that the project definition be reviewed to ensure compliance. Furthermore it is recommended and that a variety of platforms (computers, processor types and browsers) be used to check the world in order to maximise the potential audience and optimise the world performance.

Those respondents using the Workflow method expressed the view that the approach provided a sound framework for which to base world development around. It was noted however that the model indicated only the major milestones of a world building project and that this approach generally required supplementary augmentation at some of the stages.

2.1.3 The Code and Fix Approach.

The code and fix approach commonly referred to, as “hacking code” remains a perennial favourite for software developers. The approach is a simple one, construct the code, test the code and fix the errors when they occur.

Such an approach does not adhere to strict software engineering paradigms but allows the world developer to interactively create objects. The code and fix approach however is of somewhat dubious use beyond that of experimenting with, and prototyping objects within a very small development team.

The approach pays little attention to specifying world and object requirements, or planning a world design, these two stages generally being no more than a mental model of the world in the world builder's head. The iterative testing regime may seem attractive at first glance, however, experience has shown that the approach tends to test the specific change in the code and not on its impact and successful integration with surrounding code and objects.

The application of the code and fix approach how ever intuitive and attractive it may be to both the novice and experienced world builder, it is suggested, will increase world development time and detract from the overall quality of the world.

Those corespondents advocating the code and fix approach were, without exception, non-commercial developers without any formal grounding in the software engineering disciplines. It was noted with some interest that follow up correspondence with these world builders indicated that most had integrated the code and fix approach into some in house design method. It is suggested that this change occurred as the respondents became more familiar with VRML and therefore identified a need for a more formal approach to world development.

2.1.4 In house Design Methods

The greatest number of respondents used “in-house” development methodologies for VRML world construction. It should be noted that not all of these had a formal computing background the respondent each developing a construction process to suit their own level of VRML ability, background and requirements.

On analysis it was observed similarities existed between these individual approaches. Ignoring the actual specific mechanics of each of the approaches the development process was observed to broadly follow an iterative life cycle consisting of seven steps.

1. Situation Phase.

All in-house approaches reviewed started with the documentation of the “situation”, or more formally the world specification.

The quantity of documentation for the phase produced varied from a few paragraphs to a few pages of text, however, each used the situation documentation for the same purpose, to enable the developer(s) to retain a clear vision of what the world was required to contain and do.

2. Investigation Phase.

The investigation step of the identified life cycle concerns the reusability of existing VRML objects for inclusion into the world under construction.

The reuse of objects previously created, either by the world developer or acquired from other sources (many VRML object repositories exist on line, most providing royalty free objects) clearly reduce the development time for the world. It was noted, however, that reusing existing objects carries its own pitfalls, for example the world object will require a specific scale so as not to be out of place when place in context with its peers.

Existing VRML objects generally are constructed as single object worlds and therefore require no specific scaling details.

3. Design Brief Phase.

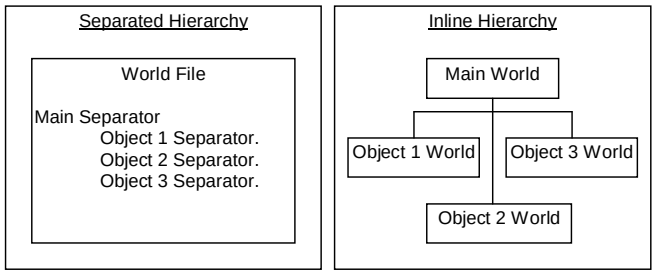
Using the situation specification and pre-designed objects from the investigation stage a Design Brief for the world is constructed.

The Design Brief details the planned structure of the web that is to be constructed to support the VRML world. Issues such as:

- How the world will be accessed, (directly, by another VRML world or via an HTML page link).
- The physical support structure for the world (where textures, images, sounds etc. are stored within the web)
- The platforms to be developed for and tested against are outlined in advance of the actual production of the world

The Design Brief also considers the structure hierarchy of the world to be developed. There are two basic methods of hierarchy catered for in VRML, separated and inline (shown in figure 3 opposite)

Figure 3: Representation of Basic VRML Hierarchies.



Of the two hierarchy structures the simplest is the separated hierarchy, essentially all objects are contained within the boundary of the world file its self delimited by the VRML Separator/Group syntax. This has the advantage of having a “complete” world

for the browser to manipulate. The separated hierarchy has the drawbacks of large file sizes (and therefore download times) and requires any objects identified for reuse during the investigation stage to be reworked to fit into the world.

The inline hierarchy utilises the VRML WWWInline syntax to break the objects of the world into individual singleton object worlds in their own right. These object worlds are connected to the main world at download. The advantages of the inline hierarchy are that it uses the software-engineering concept of modularity to aid in world maintenance, allows existing objects to be fitted into the main world without changing the object itself. The disadvantages of the inline hierarchy, however, are that the browser is in effect manipulating may worlds, potentially affecting the performance of the world and requires all object worlds to be present at download time. If the object world is unavailable it simply does not appear.

The individual hierarchies are not mutually exclusive and finding the correct balance for the world between the two approaches will minimise the drawbacks of each.

4. Modelling.

Using the information gathered from the previous phases develop prototype objects to fill in the missing parts of the world.

The advantages of prototyping at this stage are many (see Avgerou and Cornford (1993), Pressman (1994)). The primary function of the Modelling phase is to permit the world builder to explore various possibilities of the world's form and content. This is clearly vital if the developer is using the inline hierarchy approach within the world.

It is noted that the developed prototypes need not be complex or detailed in nature and that the use of modelling tools (of various disciplines, CAD tools, geometry modellers,

3D generators etc.) were extensively used by the respondents to produce rapid prototype worlds.

5. Working Drawing.

Having experimented with the form and content for the world a solution is selected for further development. The layout of the world is drawn out as the Working Drawing document.

The Working Drawing document is a synthesis of the previous phases, it is intended to reflect a graphical overview of the world, world information such as lighting and view points, world objects and their placement within the world and any physical restraints placed on the world by the Design Brief.

It was observed that there was little concordance between the respondent as to the method of actually constructing the working drawing document each using the tools with which they were most familiar. The physical medium of the document is not, therefore, restricted to physical paper, both soft copy and in one case a representational mock-up of a room using children’s plastic construction blocks were used to augment the Working Drawing documentation.

The nature of documenting a 3D representation on a 2D medium also differed with the respondent's background. This varied styles including, traditional technical drawing styles, single overhead bird's eye view plan, pseudo 3D plans using logarithmic paper and a series of different world views over a number of pages (top, front, side etc. views).

7. Realisation.

Having established the criteria for the world work the process of creating the world is undertaken.

The Realisation phase draws on all the previous stages to actually build the VRML world. From the bottom up this takes the Design Brief and places the world components and objects within the physical web structure. Using the Working Drawing documentation the objects developed and used by the Modelling and Investigation phases are used to populate the world in their appropriate places. Additional world information explored during the Modelling Stage and documented in the Working Drawing is added.

It was noted that those respondents developing using the VRML 2 specification added object and world behaviour at this point of the life cycle. It is assumed that preliminary work on behaviour must have been conducted during the Modelling stage and that this is simple the refinement and integration of object's behaviour according to the Situation specification and Working Drawing

8. Evaluation.

In order to check the “correctness” of the world against the project documentation a review stage is conducted.

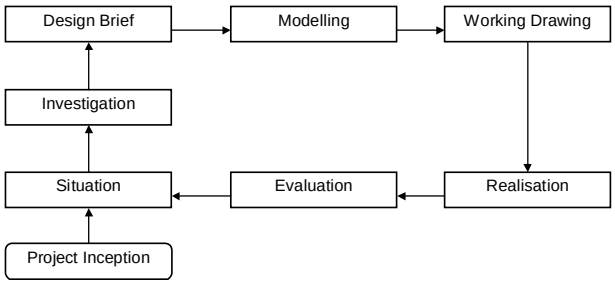
The evaluation stage combines two roles it checks the world against the Situation specification and permits the review of the Design Brief to ensure that the world is compatible with the target platform.

Those respondents developing commercial worlds indicated that at this point the completed world could be demonstrated to the client for their acceptance. It was noted that the finished product seldom matched the exact requirements or expectations (arguably due to media hyped misconceptions of virtual reality) of the client.

When such mismatches occur they can be used to refine the situation specification and embark on another cycle of the development process.

Figure 4, opposite, shows the life cycle of the in house development approach, drawn from the common threads of the reviewed individual approaches to developing VRML worlds.

Figure 4: Generic Model of "In House" Approaches.



Analysis of the four reviewed approaches, client driven, Workflow, code and fix and the generic in house approach concludes the following:

1. From the empirical evidence gathered by the survey it is concluded that the first hypothesis is true. If software-engineering paradigms existed that supported VRML development then there would be no need for Silicon Graphics' publication of their Workflow process (except for kudos). Consequently, world builders would not be developing in house design methods and commercial world builders would not be attempting to dissuade clients from utilising existing software engineering paradigms.

2. Commercial clients require VRML world building to be conducted within a frame work that is indicative of expected quality standards such as planning, audits, review point and documentation. If such a framework incorporating a supporting development paradigm does not exist then VRML will no be perceived as a quality development medium, proving the second hypothesis. The commercial respondent's experiences and the movement of the code and fix practitioners to a more structured way of development provide further supporting evidence for the hypothesis.

The conclusions naturally present the question *why* are software engineering paradigms not being used?

2.2 Existing Software Engineering Approaches.

The evidence gathered in support of the two hypotheses indicates that software-engineering paradigms are not being utilised for the development of VRML environments. However due to the relatively small amount of respondents to the research survey it can not be assumed that the evidence proves the hypothesis.

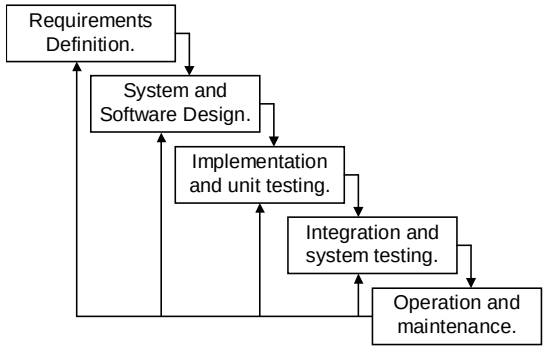
It is considered that in order to gain further support for the hypothesis that various existing software-engineering paradigms need to be reviewed in order to ascertain why they are not being widely used for VRML design. Clearly the plethora of paradigms available to the software engineer precludes the full investigation of all possible approaches. As such three popular software engineering approaches, the classic, evolutionary and object oriented will be explored and considered as to their suitability for VRML development paradigms.

2.2.1 The Classic Approach.

The classic or traditional approach is widely recognised as the oldest and most prolific paradigm of software engineering (Bischofberger and Pomberger (1990)). The process was originally derived from the contemporary manufacturing engineering disciplines in the early 1970's (Royce 1970) and rapidly became a formalised model for software development.

The model attempts to map the various aspects of a system development to a specific phase within the development life cycle. The number and complexity of each of these phases vary widely depending on which literary source is referenced resulting in some

Figure 5: The Classic or "Waterfall" Model.



confusion as to the nature of the model (Baker (1996)). The fundamental phases are, however, generally identified as; requirements analysis and definition, system and software design, implementation and unit testing, integration and system testing, operation and maintenance, as shown in the waterfall model, figure 5 above (Sommerville (1996)).

Each phase of the model identifies a specific deliverable that forms the input for the next phase in the sequence. The final stage, maintenance, feeds identified problems with the derived system back to the initial stage, thus instigating another development cycle or independent project.

Advocates of the classic approach to software engineering using the waterfall model stress that the model provides a clearly defined division of the important activities within the development process. The division provides a logical and understandable method of development that facilitates division of labour and is applicable to any development project regardless of the scope or size of the problem domain. Equally it could be also be argued that this approach to engineering is stable having been proved during its 20 year service to the industry and that subsequent paradigms are essentially only refinements or adaptations of the classical approach.

The classic approach to engineering software systems using the waterfall model has been the target for much criticism since its formalisation. The majority of these criticisms are based on the linear and sequential nature Avgerou and Cornford (1993b), identify three major criticisms of the classic development life cycle as “fallacies”.

1. The Fallacy of Accurate Specification.

The model assumes that the analysis completed during the requirement definition stage will have been able to identify all of the client's requirements. Critics state that this is highly unlikely, as clients are rarely able to provide a complete specification of requirements. Even where the analysis of the client's requirements has been thoroughly examined and identified this only represents a common understanding of the problem

domain between the client and analyst and does not guarantee a final system that will fulfil all the client's expectations.

2. The Fallacy of Linear Development Sequence.

Software development is seldom a linear process following a sequential flow of operations. Problems discovered during a phase as a result of inaccurate or missing data from a previous stage may call a halt to development while the erroneous stage is reviewed and corrected. This potentially could lead to extensive rewriting of the following phase deliverables if the problem has not been detected at a much earlier phase.

The boundaries between the different phases are not clear cut resulting in a blurring of the start/end of consecutive phases. This may lead to the start of another phase before the deliverables of the feeder phase have been finalised. Such an overlap, it is claimed by critics, will increase the chances of a phase being embarked on without complete inputs from the previous level (see above).

3. The Fallacy of the Complete System and Maintenance Burden.

The model assumes that the finished system will be acceptable to the client on delivery with subsequent modifications being made under a maintenance regime or spawning a new project. Avgerou and Cornford observe that a client’s organisation may separate development of new projects from maintenance of existing systems this leads to a reduction of the intellectual capital invested in the development process as the development team are divorced form the maintenance process. Avgerou and Cornford further observe that organisations expect to be able to extend the life of a system through modification, the model does not cater well for such a strategy of evolution.

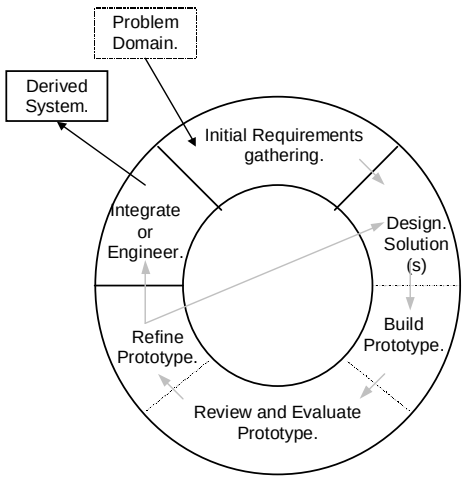
Pressman (1996b) also notes that because of the complete system being delivered as a whole a client must be patient as no working version of the system will be available until the operation and maintenance phase of the model.

It is considered that although such criticisms of the classic development life cycle are valid, the approach has been in existence longer than rival methods and so has had more opportunity for critical scrutiny. Further it is observed that the environment for which the model was initially developed, that is the mainframe area of computing, has declined with the advent of the desktop PC and rapid software development tools. It is therefore suggested that the software development community has attempted to bend an acceptable paradigm to a different use than that for which it was intended, effectively putting new wine into seasoned wine skins.

2.2.2 Evolutionary Development Approaches.

The fundamental philosophy to the evolutionary approach is one of developing a solution, reviewing the solution and refining the solution. This is generally achieved through the process decomposing the problem domain into areas of functionality, combining the requirements, design, implementation and testing phases and developing a prototype solution for the specific area identified (See figure 6).

Figure 6: The Evolutionary Model



Using the evolutionary philosophy both developer and client explore the problem domain in order to discover the systems requirements. Each functional unit is examined in order to derive a requirement specification or to further decompose the unit into parts.

Using the unit specification a solution is designed code generated to support the requirements. Both the developer and client can determine if all identified requirements have been met and propose additional requirements review the realised solution.

If new features are required or identified functionality has not been realised the unit is refined by re-iterating through the cycle. If both the client and developer are confident in the derived solution two possible courses of action are available (Brooks (1975)):

1. Integrate the Prototype.

   The ability to integrate the prototype into the system proper will depend largely on the functionality and quality inherent in the design process. If the module has been proved to be stable and designed with integration in mind this may pose little problems, however if the prototype has been a "quick and dirty" solution this may be an unwise path to take.

2. Discard the Prototype.

   Having explored the required functionality to a satisfactory state the prototype has effectively outlived its usefulness and is thrown away. Using the identified requirements from the prototyping exercise the module can be written using software engineering principles for integration into the system.

The distinction between these two routes is subtle but has potential differing uses. Discard allows the developer to concentrate the client on poorly identified requirements by producing a rapid succession of prototypes but has the draw back of having to create a module from scratch. Integration provides a slower development track as the client and developer "grow" the module together but results in a module ready to "plug" into the system.

The advocates of the prototyping approaches to software development stress the improvement of communications between the client and the developer. The enhanced information flow between these two collaborators has a number of ramifications.

1. Risk and uncertainty is reduced as both collaborators can determine the unspecified functionality of the system and formulate a solution that is both acceptable and appropriate to both parties.

2. Time and expense of the system development is reduced, as problems can be resolved during the prototyping stage. A study by Boehm et al (1975) have shown that utilising the prototyping approach to system development cost 40% less and require 45% less effort. Similar studies by Scott (1978), Berrsiford and Wether (1979), Mason and Cary (1983) and Bonet and Kung (1994) all concur with the Boehm hypothesis and conclusions although empirical evidence of the savings differ.

3. The client is more likely to accept the finalised system evolved by either of the prototyping approaches because of familiarity to the system. This principle in effect reduces the amount of "culture shock" that is attributed to the newness of the system.

4. The costs of all stages are reduced as the client shares the development costs in proportion to the level of interactivity afforded by the approach. The temptation to augment or "gilt edge" the system with redundant or inappropriate additions is moderated by the participation of the other collaborator. If both collaborators agree then there is the opportunity to further develop the system.

5. The management of the project is simplified by the partitioning of the system into functional modules by applying a "divide and conquer" principle. It is argued that decomposing the proposed system enables the estimates, scheduling and budgets to be identified and assigned at an earlier point in time, therefore critical path analysis, and risk associated with change in circumstances (such as budgetary cuts, changes in specification etc.) can be planned for better.

The evolutionary paradigms are not without their problems, however.

1. The approaches rely on the ability to rapidly produce a software prototype if this can not be realised by the constraints of the development team resources, man power, CASE

tools, rapid development tools (4th generation tools) or familiarity with such tools then the benefit of the rapidity is lost to extended development time.

2. The evolving system by use of prototypes may lead to a misunderstanding between the developer and the client. By their nature prototypes are imperfect and illustrative, this may undermine the confidence of the client in the programmers capability as they may have limited functionality. Conversely a client who continually changes the requirements in order to explore the possibilities will limit the developers opportunity to impart functionality to the prototype.

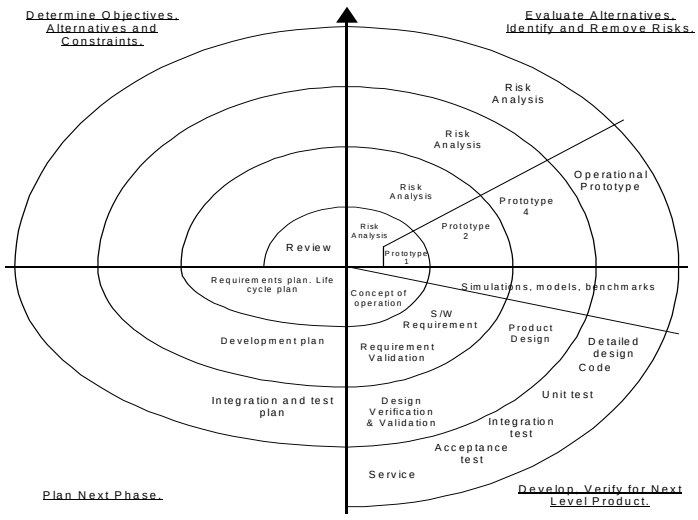
3. The approach relies heavily on the social engineering and communications skills of all collaborators within the project. It assumes that the common conduit for information flow between the collaborators is at a level obtainable by each collaborator. If there is a mismatch in expressive and comprehensive ability then the risk of misunderstandings between and repression of one or more collaborators is increased.

4. Finally the prototyping approach to evolutionary development may not be suitable to one of the collaborators. This is exemplified where the client is used to a more traditional approach to system development under a classical waterfall development regime or requires definitive task be completed in order to fit the software quality assurance plan, for example.

In order to overcome some of the prototype approach limitations Bohem (1988) proposed a new development model combining the benefits of the classical and prototype approaches, this model is widely referred to as the “spiral model” of software development.

The spiral model (see figure 7 opposite) divides the system development project life cycle into four distinct areas

Figure 7: The Spiral Model.



1. Planning.

Determine the objectives, alternatives and constraints of a proposed system.

2. Risk Analysis.

Evaluate alternatives. Identify and resolve risks in developing the proposed system.

3. Engineering.

Develop and, verify a solution for the problem.

4. Evaluation.

Plan the next phase of the system development from the client response to work completed to date.

Starting at the centre of a spiral the areas are moved through each providing it's deliverable product as an augmentation to the larger system. As the different areas of development are moved though the spiral is widened thus iterating through the four areas but building on the experience of previous iterations thereby working toward a more complete system.

The spiral model is widely taught and recognised as being the most realistic model of software development within the context of project management. The evolution of the system is not totally reliant on the use of prototyping, which is used as a risk reduction strategy, and retains the familiar systematic life cycle of the classical approach. By it's iterative nature the spiral

model removes the problems of the linear start/stop development of the waterfall model by naturally progressing the development through the stages of prototyping.

2.2.3 Object Oriented Approaches.

The concept of object oriented perception was first muted by the ancient Greek philosophers. The philosophy reasons that the universe is made up of many objects that interact with each other. Objects can be grouped according to their nature, size, colour, interactive behaviour with other objects etc. An object either is composed of a number of subordinate objects each with its own nature or "properties" or exists as a "primitive" object that can only be used to construct other objects from.

Descartes, the seventeenth century philosopher observed that human beings naturally adopt an object oriented view of the world and in understanding it's processes, an observation continued in the research of modern philosophers such as Minsky (1986) and Rand (1979). The object oriented approach as a software-engineering paradigm presents this "natural" affinity to viewing problems as the key to deriving solutions to complex problem domains.

The object oriented model assumes that it is possible to decompose the problem domain into a less complex series of interrelated objects. This decomposition is afforded by the principles of abstraction, encapsulation, modularity and hierarchy. Booch (1996) formally defines these principles as,

1. Abstraction.

An abstraction denotes the essential characteristics of an object that distinguish it from all other objects and thus provide crisply defined conceptual boundaries, relative to the perspective viewer.

2. Encapsulation.

Encapsulation is the process of compartmentalising the elements of an abstraction that constitute its structure and behaviour. Encapsulation serves to separate the contractual interface of an abstraction and its implementation.

3. Modularity.

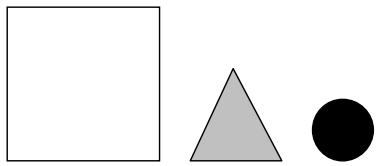
Modularity is the property of a system that has been decomposed into a set of cohesive and loosely coupled modules.

4. Hierarchy.

Hierarchy is a ranking or ordering of abstractions.

Less formally, abstraction identifies characteristics of an object, encapsulation implements those characteristics, modularity asserts the independence of the object and hierarchy presents a framework to manage the objects by. Objects are seen as holding both data and behaviour (properties and methods) and either exist in their own right or are constructed from a template archetype object or class, which may in tern utilise the resources of other objects.

Figure 8: Objects A, B and C.



By way of example consider the problem of realising three objects, A, B, C as shown in figure 8 opposite. Applying the principle of abstraction the unique characteristics of the objects are observed as object A is a large white square, object B is a medium sized grey triangle and object C is a small black circle.

Utilising the characteristics defining the objects the process of encapsulation can be applied. Each of the object characteristics can be categorised. Object A has a shape (square) a colour (white) and a size (large). Object B has a shape (triangle) a colour (grey) and a size (medium). Object C has a shape (circle) a colour (black). The rules governing the geometry of shape construction, constraints of size and colour calculation can be formulated for each of the given objects. Encapsulation provides a path to data hiding the three objects remain the same

externally as defined by their characteristics, a large white square, a medium grey triangle and a small black circle, but each object now has the hidden information to recreate it encapsulated in itself.

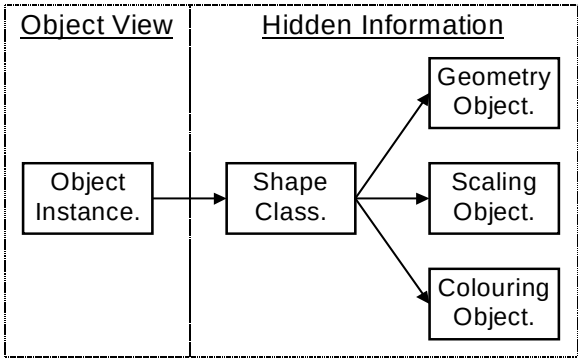
Modularity requires that a system be broken down into a series of individual objects. An object's quality can be assessed by the degree of functional independence it exhibits through its coupling and cohesiveness. Cohesion is a measure of functional strength of an object, coupling is the measure of relative independence among other objects. The software engineering philosophy regards high cohesion and loose coupling to be optimal. The objects A, B, and C can be said to have low cohesion as each performs more than one task in order to be realised, calculate geometry, calculate scale colour bounded area. The objects may be considered as having no coupling as there is no interconnection between, and therefore no interaction between these objects.

In order to increase cohesion and establish coupling the encapsulation process requires reviewing. From the presented objects it can be ascertained that there is commonality between the objects implementation. Each object implements a geometric calculation a scaling calculation and a colour calculation. Removing these from the objects it is apparent that the three objects are in fact one object, shape with the single function or method which displays the shape and holding the properties of form, scale and colour. The geometry, scaling and colouring functions can be realised as abstract objects within their own right interacting with the single shape object by means of the properties. This single shape object becomes the template or class for all three instances of shape, objects A, B and C.

The new definition of the objects leads to further information by encapsulation, increased cohesion as the class shape performs only one function of display shape calling on the three new objects geometry, scale, and colour to realise the display task, therefore becoming coupled (in this case loosely).

To realise the objects A, B and C a hierarchy needs to be established in order maintains order between the different modules of the system. An object (A, B or C) is an instance of type class shape, in order for the object to be realised the class shape calls upon the three abstract objects to perform the functions of geometry, scaling and coloration from the object's held properties. The hierarchy can therefore be constructed as shown in figure 9 opposite. Note that the view of the objects remains the same irrespective of the additional hidden information of encapsulation, modularity and hierarchy that provides the view.

Figure 9: Module Hierarchy for Objects A, B and C.



The object oriented model has found favour within the different divisions of the software engineering discipline being applied to analysis (OOA), design (OOD) and programming (OOP). The approach provides an iterative approach to problem solving. Indeed it is considered by many to be a refinement of the evolutionary approach although it is respectfully suggested that the basic philosophy underlying the object oriented approach differs to that of evolutionary approaches clearly identifies it as a unique approach with a similar implementation.

The approach provides some clear benefits:

1. Modules (collections of objects and classes) can be designed and reviewed independently of each other.
2. Modules can be reused either within the system or incorporated into other systems (there is no need to reinvent the wheel).

3. Management of the project becomes simpler as each module can be created, validated, maintained and documented individually providing a degree of security and resource control.

Critics of object oriented approaches determine three failings, understandability, applicability and support methodologies.

1. Understandability.

The software engineering object-oriented paradigm is relatively new, being formulated in the late 1980's. It is argued that such a radical approach, no matter how intuitive, is impractical due to the investment (intellectual, financial and temporal) in "traditional" engineering processes. To realise the object oriented approach would require a massive retraining of software engineers, re development of existing systems and fundamental change to the principles of software engineering it's self.

2. Applicability.

It is observed that not all software-engineering problems can be solved using the object-oriented paradigm. While it is conceded that using an object oriented approach may produce a series of modules it is simpler in some instances to revert to traditional engineering practices to connect, control and derive those modules, why produce a network of objects when a short segment of "traditional" code is sufficient?

3. Support Methodologies.

The relative newness of the object-oriented approach, it is conjectured, means that it is unproved. Support methodologies proposed by advocates such as Booch (1996), White (1996) and others are still in a process of evolving and therefore can not be relied on to produce an accurate refection of the object oriented process. Such methodologies are further criticised for the complexity of notation required in the construction and conveyance of system concepts.

Having reviewed the three software engineering approaches an analysis of the question “why are software engineering approaches not being used?” observes the following conclusions;

1. Software engineering paradigms are not being utilised wholesale for VRML world construction because they are not seen as being applicable to the medium. VRML is not widely perceived as a programming language rather a method for generating graphical images.

2. Limited use of an evolutionary style of development using prototyping techniques is being employed by at least two of the approaches reviewed in the world builder survey, Workflow and the generic In House approaches.

3. The variety of background’s that compose the VRML world building community (based on evidence from the survey of world builders as discussed in section 2.1) means that non computer discipline world builders may have no, or little, exposure to formal software engineering paradigms, therefore such paradigms are not being employed.

4. Because VRML is perceived as a graphic *tool* and not a programming language there is a resistance to the use of scientific engineering principles as they are seen to restrict the “natural” artistic ability of the world builder. It is noted that this is not restricted to VRML world building but is an attitude prevalent within the computing industry at large. This is a basic misconception of what a methodology, paradigm or model represents, a framework to work *within* as opposed to a structure to slavishly conform *to*. (See Fitzgerald (1996) and Wastell (1996))

Comparing the software paradigms reviewed with the respondent's methods it is clear there is a mismatch between the needs of the artist using VRML to express their creativity by the medium of the computer and the software engineer using VRML to construct graphically oriented software solutions. What is required is an approach to VRML world building that is acceptable to world builders at *either* end of the development spectrum.

2.3 Existing Tool Support for VRML World Building.

The conclusions thus far indicate that although software engineering paradigms are not being used per se as methods for VRML world development, the Workflow and In House approaches do utilise the principles of the evolutionary approach by the process of prototyping.

As previously discussed prototyping is heavily dependent on the ability to rapidly produce system components in order to ascertain their worth. Such rapid realisation of a system is consequently dependent on the level of tool support for the development process. It is, therefore, considered vital to understand and review the tools that facilitate the realisation of VRML worlds. This section intends to present an overview of the generic tools that are available to the world builder

2.3.1 VRML Browsers and Viewers

The browser application provides the interface between the world participant and the VRML world. In general a browser provides a number of ways for the world participant of examining and navigating through the scene presented by the VRML world file. The actual implementation of participant interaction control differs from browser to browser but it is observed that most offer similar functionality, such as examine mode (allowing the participant to rotate around a specific world point or object and a fly mode (providing a full six degrees of freedom to move through the world). Browsers generally include additional controls such as, walk mode (a refinement of the fly mode allowing the participant to move through the world on a 2D plane), seek mode (allowing the participant to select a world point or object toward which the viewpoint is moved).

Browsers have been implemented in one of two forms, either as a stand-alone application (some times referred to as a VRML viewer) or as a “plug-in” extension to another software application, usually an HTML browser such as Netscape or Microsoft Internet Explorer.

Stand-alone viewers are written to be able to retrieve, and interpret VRML code using the standard MIME Internet protocol. Stand-alone viewers have the advantage of being independent software entities there by allowing selection to be made on VRML version (1 or 2), vendor specific implementations of protocols interface and specific needs.

Plug-in browsers are somewhat simpler working in conjunction with the host program to retrieve and interpret VRML files. Plug-in browsers have an advantage in the fact that they allow the integration of both VRML and HTML (and other Internet protocols) into one package using similar protocols.

The selection of browser by type is a matter of personal choice as both achieve the same result although it is noted that there are relatively few browsers that implement the whole of the specification options for either VRML versions 1 or 2.

2.3.2 VRML File Editors.

VRML world files utilise the ASCII character set, either directly for version 1 files or as a subset of the UTF-8 character set for version 1.1 and 2 files as defined by the ISO 10646-1:1993. In order to create and edit VRML world files any UTF-8 compliant text editor available to the world builder may therefore be used providing that the file is saved in the UTF-8 format with the common world extension of .WRL.

Most operating environments provide such a text editor as default, UNIX VI or sed editors, MS-DOS edlin or edit, Windows 3x notepad, Windows 95 and NT wordpad (saving the file as plain text) etc.

A number of VRML oriented editors are available to aid the world builder with the task of constructing the world file. These editors provide additional functionality over standard text

editors with the inclusion of syntax checking, template code sections and text formatting to assist in the readability of the code within the environment.

2.3.3 VRML World Editors.

World editors are a logical extension of the text based VRML file editor within the graphical context of the VRML medium. World editors replace the text entry interface with a graphical interface allowing the world builder to effectively draw the world and its component objects on screen.

In general world editors provide the world builder with a blank world space, a set of primitive VRML objects and a tool kit of functions such as texture mapping, geometric manipulation and lighting and camera set-up. Because editing of the world is conducted graphically within a virtual or semi-virtual space the world builder is able to immediately assess the impact of changes to the world scene as they are made. Clearly this is a distinct advantage over file editors, which must saved and then view files through a browser in order to assess any change.

It is observed, on the basis of the world editors reviewed, that while there are clear benefits associated with using world editors, rapid assessment of changes, “intuitive” construction of objects and worlds, disposable prototype generation etc. there are a number of drawbacks. The notable disadvantages observed are:

1. VRML is not the native file format of such editors with the VRML world being exported on completion. This precludes the VRML file being used for refinement, rather the saved native format file is edited and re exported. This has clear implications for world component reuse.
2. The process of exporting to a VRML format is generally approached utilising a polygon per face strategy. Such a process creates large and unoptimised worlds containing multiple indexed pint and co-ordinate sets that take longer to render than

simple primitives do and reduce the readability of the file and any subsequent maintenance by the world builder.

2.3.4 VRML Support in Other Applications.

It is observed that the formalisation of the VRML specification has lead to the acceptance of the format by non-VRML graphic software developers. As such a number of applications including CAD software, geometry modellers, 3D design tools and image renderers with 3D awareness contain the option to export their products to VRML format.

While the similar benefits and restrictions to world editors apply it does mean that potential world builders who are familiar with such applications do not have to retrain to other packages with a more direct VRML support.

2.3.5 VRML Converters.

A number of file format converters are available to the world builder. In general these converters are stand alone software applications which translate a specific file format to a VRML compliant file. These include the generic 3D .DFX AutoCAD format, the .WAD format popularised by first person games such as Doom, ray tracing file formats .POV, .RAW and .NFF to name but a few.

Arguably, the most useful of these converters is the VRML version 1 to version 2 format converters. The use of such a converter not only allows the rapid “upgrading” of existing version 1 .WLD files to the version 2 specification, but also provides the world builder with the ability to produce rapid visual prototypes in VRML 1 (a simpler language than VRML 2) for client assessment. Such prototype worlds can then be converted to the VRML 2 specification for additional and interactive augmentation.

The rise of tool support for VRML world building represents a double edged sword. On the one hand the increased speed that a world can be generated and viewed means that VRML has the potential to be truly realised as a development medium for interactive graphical applications. On the other, the increased access to tools that facilitate world production emphasises the need for their use within a frame work to ensure that some form of quality assurance can be afforded to both the commissioning client and the eventual world participant. With power comes responsibility.

The producers of VRML viewers and browsers further complicate this issue. It is observed from the research conducted into tool support that the implementation of either of the VRML standards is not being conducted consistently by browser developers. Functionality that may exist within one browser application may not be present or implemented differently in another. Typical inconstancies include, view points change, rendering strategy differences, a particular image format may not be valid or not mapped in the same manner and more dramatically a world that is fully functional in one browser may not work or even download into another. While it is conceded that many of the VRML browsers reviewed are so called Beta releases the different approaches to the same task does little to aid the task of the world builder.

2.4 Limitations Research Conclusions.

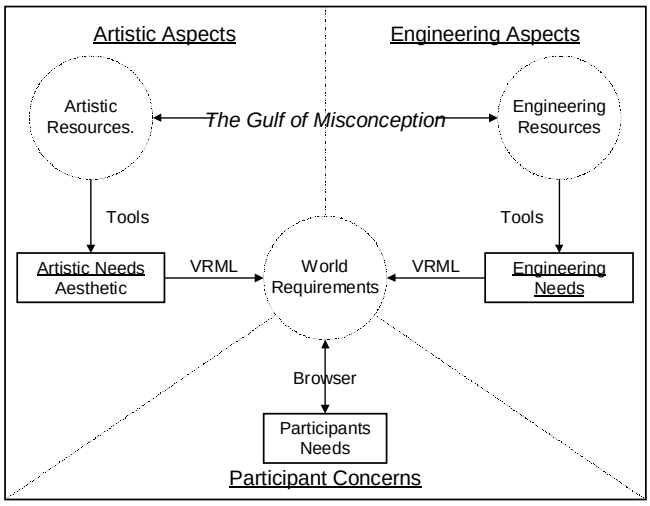
From the research presented in this chapter it is concluded that the first of the initial hypothesis presented in chapter one is true to some extent, current software engineering paradigms are not well suited to the development of VRML worlds. It is also concluded that the second hypothesis that unless some form of recognised method for VRML world development is not used then world participants (and commissioning clients) will not have confidence in the quality of VRML as a medium for the graphical presentation and dissemination of information.

It is suggested that underuse of software engineering paradigms in VRML world building can be directly attributed to the different perspectives held on what the VRML medium actually represents by world builders.

1. The artistic or *aesthetic* aspect perspective of VRML as a tool to realise art through the medium of the computer.
2. The engineering or *methodical* aspect perspective of VRML as a tool to realise software implementation graphically.

This "gulf of misconception" between the artistic and engineering aspects of VRML world building must be addressed if the concerns and expectations of the world participant are to be realised successfully (see figure 10 opposite).

Figure 10: The VRML Gulf of Misconception.



It is noted that the disparity of the artistic and the methodical aspects of HTML web page construction have lead to an increase in the “junk” pages and hypertext links that currently litter the Internet. It is suggested that this phenomena is directly attributable to the increased access to HTML construction tools by those who lack an appreciation of these disparate aspects or who have had no guiding method to address such failings.

It can not be expected (and indeed it *should* not be expected) that the artistic nature of the VRML medium should be made to conform to a scientific methodology for such would inevitably restrict the graphic artists creative ability. Conversely sound methodical engineering approaches to world building must not be abandoned for such will inevitably lead to poor quality of world performance.

In order to bridge this gulf of misconception a synthesis of the two different aspects that is acceptable to both the artistic and engineering needs of the medium is required. It is postulated that until such a holistic approach is formed encompassing these two aspects, then the concerns of the world participant can not be addressed in a full and meaningful way. The much lauded adage “build and they will come” does not hold true, the goals of VRML are not realised, the “cyberspace” vision fades.

3 New Ideas.

"The use of 3D (and possibly immersive) interfaces will change the way software engineering is done, but it's hard to predict at this early stage just what those changes will be."

Bernie Roehl (1996).

The preceding chapters have presented and drawn conclusions for two hypotheses regarding the applicability of existing software-engineering paradigms to the development of quality VRML worlds. It has been shown through empirical evidence and analysis that such accepted paradigms are under used and only partially applicable to VRML development, consequently the quality of world development using VRML under such paradigms is questionable.

From the presented evidence and conclusions it is considered that a *new* methodology for the creation of worlds that provides a "systematic pattern of all actions necessary to provide adequate confidence that the item or project conforms to established technical requirements" (Manns and Coleman (1996)) is required.

If such a method is to be developed it must provide a bridge between the artistic *and* engineering aspects of the VRML medium and reduce the gulf of misconception identified as a limiting factor in current world building. Further more such a method must provide a structure for the three core activities of the software development project, Development, Management and Quality Assurance to be conducted within.

3.1 Requirements of a Methodology.

Before the new methodology can be proposed it is considered necessary to understand what constitutes a "good" methodology. Research indicates three distinct schools of thought on this subject, each concentrating on a particular aspect of methodologies, the philosophy (or Objectives) of a methodology, the content (or Characteristics) of a methodology and the Expectations of a methodology.

It is observed that within each of the three aspects there is a degree of overlap between the issues considered. While each of these aspects are valid in their own right, it is suggested that the evolution of a method from a notional requirement *must* involve all three of these aspects if it is to be of significant worth.

3.1.1 Objectives.

The philosophy of the "methodology movement" (Avgerou & Cornford (1993)) requires the consideration of a number of factors in order to justify the existence of a methodology. Simply put, the objectives of a methodology are to provide knowledge that is understandable, shareable and logical to derive a solution to a specific problem. The objectives of a methodology are not concerned with the actual mechanics of how a solution is derived through the implementation of the method but rather the *philosophy* of issues concerning the application of a methodology.

Before any work on formulating a new method can be conducted it is imperative that these concerns are addressed, if they are not then the derived process will not be accessible to or unusable to anyone besides the originator. Asking the following questions can identify such concerns,

1. Why is the method being developed?
2. How should the method implement the process of development?

3. How will the method permit management of the development project?

4. How can the processes of the method be represented?

5. What is the best way to pass on or teach the method?

3.1.2 Characteristics.

Having established the driving philosophy behind a methodology a more specific analysis of the factors characterising the methodology can be addressed. This may be seen as a high level view of the methodologies *mechanics* derived by analysis of the methods fundamental objectives.

The implementation of the objectives as characteristics, therefore, serves to identify and formulate the boundaries and constraints of the method. DeMarco (1978) identifies a good methodology as being characterised by a number of factors as follows.

1. It provides concise and complete specification of how it should be used (Objective 4 and 5).

2. It utilises an easily understood and graphical notation (Objective 4 and 5).

3. It is partitioned into individual processes and specifications that contribute to the whole (Objective 2 and 3).

4. It has a structure that allows a smooth and logical progression between these processes (Objective 2).

5. It allows for iteration within the structure so that the previous process may be reviewed (Objective 2).

6. It is maintainable allowing for changes within the specification or process to be made without impacting on other existing processes (Objective 3).

7. It provides a deliverable at the completion of a process. DeMarco suggests that this should be a paper model of the proposed system, clearly with modern rapid development tools this would suggest the inclusion of software prototypes as well (Objective 3).

3.1.3 Expectations.

In order to realise the characteristics of the methodology the mechanics of each identified characteristic must be actinide. This refinement of the characteristic processes can be seen as the low-level view of the method's mechanics, the "nitty gritty" implementation of the method, or to use the military parlance the "melee" of the methodology.

In order for each of the identified processes to be acceptable they must be able to meet a range of criteria or expectations. This becomes significant when the method becomes decomposed into a number of components contributing to the whole, as the method will be only as stable as its weakest point. Sommerville (1996) states for a process to be considered as acceptable the expectation of a process must satisfy the following

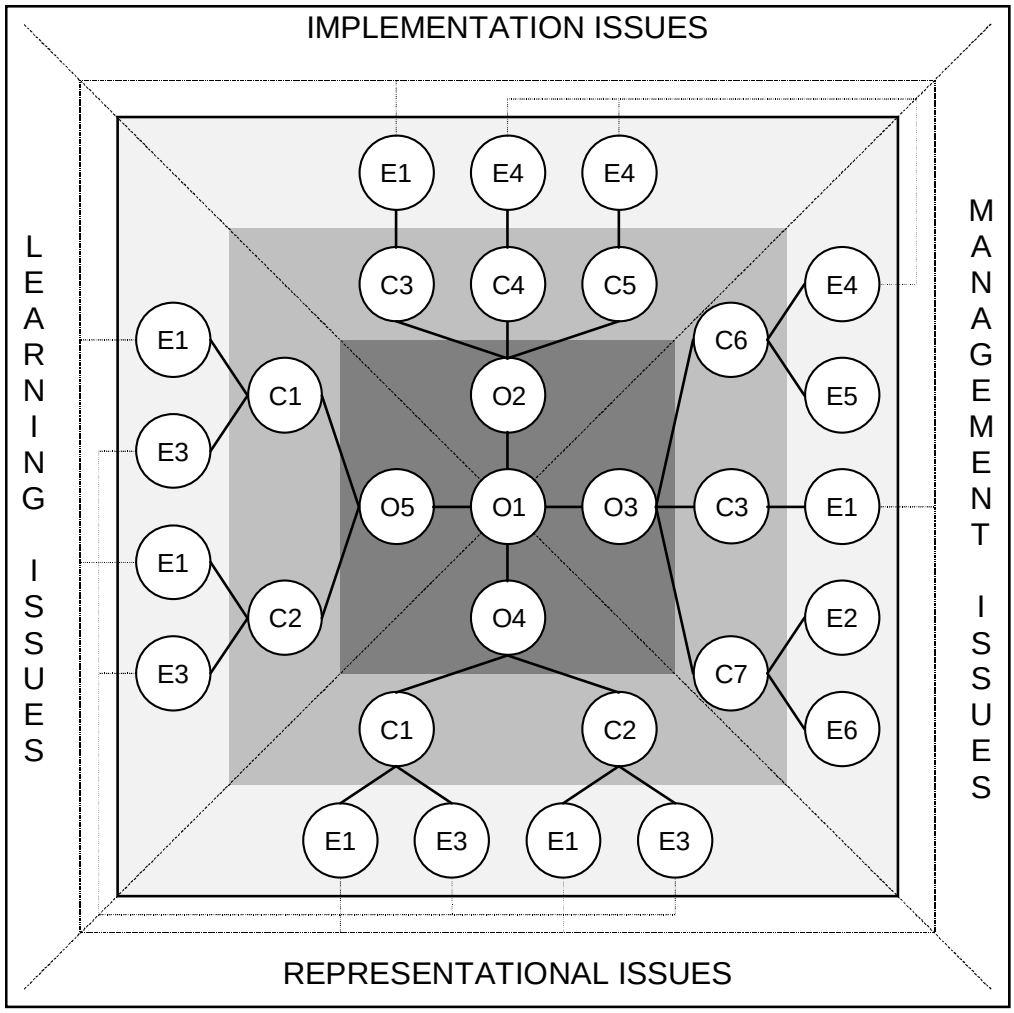
1. It must be understandable providing a clear statement of the aims and boundaries of the process (Characteristic 1, 2 and 3).
2. It must provide a visible output in a form that can be appraised by both client and management (Characteristic 7).
3. It must be acceptable (considered suitable) to all parties involved with the project (developer, project manager, quality assurance manager, client etc.) (Characteristic 1 and 2).
4. It must make provision for trapping errors before the deliverables are progressed (Characteristic 4, 5 and 6).
5. It must be maintainable and able to accommodate changes in specification and requirements (Characteristic 6).
6. It should allow the deliverables to be realised rapidly. Sommerville notes that this rapidity of development implies that the process should be supported be some form of CASE (Computer Aided Software Engineering) tool or some other form of asistative software this is clearly critical if prototyping is to be used (Characteristic 7).

If this apparent evolution through concept to realisation is, as suggested, vital to the construction of a comprehensive method then clearly an interconnection of issues is formed. This network can be constructed with the pivotal question of why the method is required (Objective 1), if this question can not justify the expenditure of effort required to realise the method then there is simply no purpose in the construction of the method for its own sake.

Assuming that there is justification for a new methodology, identified by empirical evidence and analysis of existing paradigms suitably etc, then four distinct issues are raised by the remaining objectives of the methodology: Implementation, Management, Representation and Learning. Each of these issues surrounding the justification can be addressed by identifying the required characteristics of the method and refining these by the expectations of the methodology.

Thus it is observed that the construction of the method can be afforded by a logical building and refining from first principles addressing all the noted issues and concerns. This network of issues is graphically presented in figure 11 below.

Figure 11: The Network of Interconnecting Methodology Issues.



From the representation of the issues network it is noted that three trans-issue connections are made,

1. Management and Implementation issues by the provisions made for error trapping (Expectation 4).

2. Learning and Representation issues by acceptability and suitability (Expectation 3).

3. All four issues by the requirement that their processes are clearly defined in order to be understood.

It is suggested that if it is possible to identify and address all of factors contributing to the method issues then a methodology can be said to impart significant value to the method user. If however if one or more of these factors can not be addressed with reasonable confidence then the derived method will not be balanced in one of the issue areas and consequently it's value must be questionable.

3.2 The Philosophy and Derivation of the New Methodology.

Taking the points identified in the previous section and noting the research and conclusions of chapter two a methodology to support the VRML world building process is proposed. In order for such a methodology to be derived it is considered that all points must be released otherwise the method will be incomplete and will be of limited use to the VRML world builder and not allow significant quality to be built into the project.

3.2.1 The Primary Objective.

In order to justify the proposed model as being required and not purely an academic exercise the pivotal question of why the model is required must be asked (Objective 1). The answer to this question is provided by the empirical evidence and conclusions formed in the previous chapter, there is no established method for building VRML worlds that is acceptable to the needs of the aesthetic and methodical aspects of VRML world building. In order for commercial acceptance and world participant confidence such a lack of a formal method of VRML world building may detract from VRML being viewed as a quality medium for information dissemination.

3.2.2 Implementation Issues.

From the research conducted into current approaches to developing VRML applications and software engineering techniques it is considered that an approach which is objected oriented, and allowing iterative refinement be developed. It is proposed that the methodology support the evolutionary development of a VRML world by the movement through and between a series of logical and identified phases or points. In order to provide the maximum amount of modularity and flexibility the favoured structure of the VRML world will utilise the inline hierarchy strategy (Objective 2).

It is observed that the existing paradigms for software development (both formal and VRML In-House) have a number of common elements:

1. The need for a requirements specification
2. A process for developing the product
3. A point at which the product is realised
4. A phase where an evaluation of the product can be held.

It is further observed that the process of VRML world building requires a similar number of points to be reached in order to produce the world.

It is noted that the production process, usually characterised by a coding phase, has two distinct and separate concerns with regard to the graphical nature of the VRML language, the *appearance* and *behavioural* aspects of a world and its component objects.

This presents an interesting dilemma as to which of these products is required to come first, will the behaviour modify the appearance of an object or will the appearance of the object define it's behaviour? Because of the inline structure hierarchy (worlds within the world) advocated by the proposed methodology this problem can be seen as the behaviour world of an object controlling

the object's appearance as an inline. This therefore suggests that any single world may comprise of two distinct worlds, the behaviour world and the appearance world. In order to avoid confusion as to which of these three possible worlds is being discussed in this paper the term *holding world* will refer to the object under consideration and the behavioural and appearance forms for that world will be referred to as *node worlds* or simply nodes.

It is therefore proposed that the method be broken into five discrete operations or *points*, Requirement, Appearance development, Behaviour development, Realisation and Evaluation (Characteristic 3).

In order to understand these five point's roles, with regard to the implementation of the method, a clear statement of what is expected to occur at, and what the domain boundaries are of a particular point (Expectation 1).

1. Requirement Point.

The starting point of any project requires a process of gathering information about the desired end product. The development of VRML worlds is no exception to this general observation, it is essential that the purpose and content of the world be clearly expressed, to fail to do so invites errors, misconceptions and disorder into the development process.

The Requirement Point provides the mechanism by which world requirements can be investigated, specified and documented by the developer and the client. These requirements consist of both global issues, lighting camera viewpoints etc. and world component issues such as object appearance and behaviour.

In addition to the client world requirements there are a number of external factors that must be taken into account that will have an influence on the development project. Such concerns include choice of browser, platform developed on and for, availability and use

of development tools, staffing levels and competence. These factors must be agreed on or planned for before the project commences if such a ratification is not conducted then the created world may have unpredictable results, as noted in chapter 2 or the project time scale and budget may overrun.

The goal of the Requirement Point is to derive as much information as possible with regard to the restrictions imposed by the client's expectations of the world (its behaviour and appearance) and the availability of resources to the project (tools, platforms, staff etc.).

2. Behaviour Point.

The Behaviour Point of the method represents the first of the two development points within the model. Behaviour can be expressed as two different forms:

1. *Static* behaviour, as supported by VRML 1. This includes built in behaviour nodes Level of Detail, WWWAnchor, Switch etc.
2. *Dynamic* behaviour, as supported by Java under VRML 2. This includes sound support, object manipulation, movie texture map support, the Sensor nodes etc.

The goal of the Behaviour Point is to realise the optimum behavioural form of a world in order to satisfy the behaviour node requirements identified at the Requirement Point.

3. Appearance Point.

The Appearance Point of the method represents the second of the two development points within the model. By its very nature the VRML medium is defined by the ability to graphically realise worlds and objects in 3D space.

The Appearance Point does not purely limit its exploration of the graphical possibilities of the provided primitive shape nodes but presents the world builder the opportunity to explore possibilities afforded by object manipulation nodes. Additionally the Appearance Point permits the experimentation with colour assignment and external image file mapping in order to achieve the desired affects for the world or world component.

The goal of the Appearance Point is to realise the optimum combination manipulation and construction of a of a world in order to satisfy the appearance node requirements identified at the Requirement Point

4. Realisation Point.

The realisation of a world requires the combination of the two distinct development points within the project, the behaviour node and the appearance node. The Realisation Point provides the developer with the opportunity to experiment with the interaction between both the object behaviour node and appearance node and with the interaction between the world and its objects.

The goal of the Realisation Point is to realise the optimum combination of the Appearance and Behaviour Point products in order to satisfy the world requirements identified at the Requirement Point.

5. Evaluation Point.

In order to ascertain whether or not the deliverables from a point have provided enough product to progress to the next stage. The Evaluation Point provides such a process by reviewing the point product against the expected deliverable for that point. If the criteria have been successfully met then progress to the nest point is possible. If the evaluation indicates that the deliverable is incomplete is some way then it must both be reworked

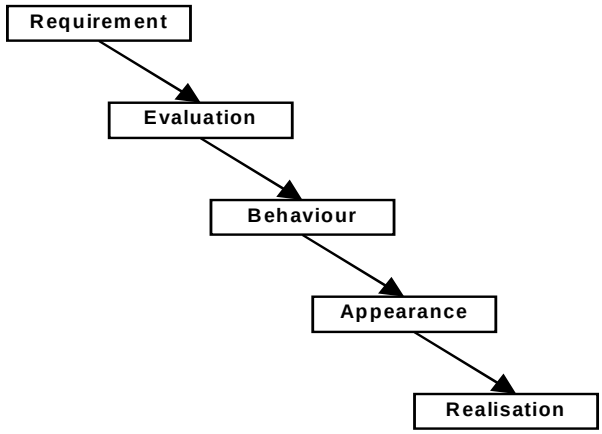
and re-evaluated or the missing criteria documented as not completed before progression is possible.

The goal of the Evaluation Point is to ensure that quantity and quality of a point product is sufficient to satisfy the world requirements identified at the Requirement Point in order to advance the development process.

Having defined the activities and boundaries of each of the individual five points of the proposed method in relative isolation a structure that permits a smooth progression between them must be constructed (Characteristic 4).

Traditional development paradigms advocate the sequence of progression as requirements, development (behaviour and appearance), realisation and evaluation, however as already noted there is a call for evaluation at all points of the development. When this is considered in the light of providing means for trapping errors at the earliest possible point within the sequence a new order of sequence can be derived (Expectation 4). The new sequence, therefore, follows the points: Requirement, Evaluation, Behaviour, Appearance and Realisation as shown in figure 12 above.

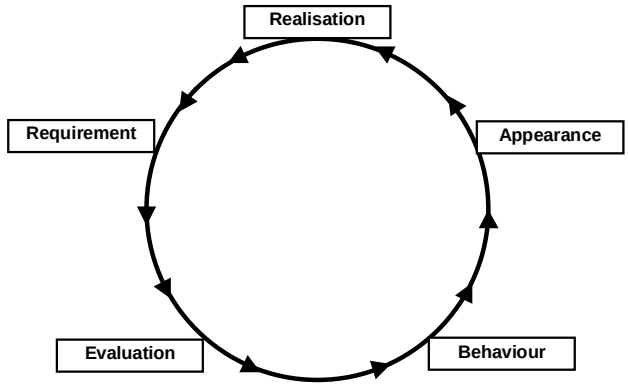
Figure 12: The Development Point Linear Cycle



This sequence may appear to be somewhat presumptuous as if followed as a single-track progression of points as it infers that if the requirements can be defined and evaluated as being correct and complete then subsequent development will be correct. It is also noted that such a

purely linear sequence does not provide the ability for each point following the Evaluation Point to be assessed as required.

Figure 13: The Development Point Recursive Cycle

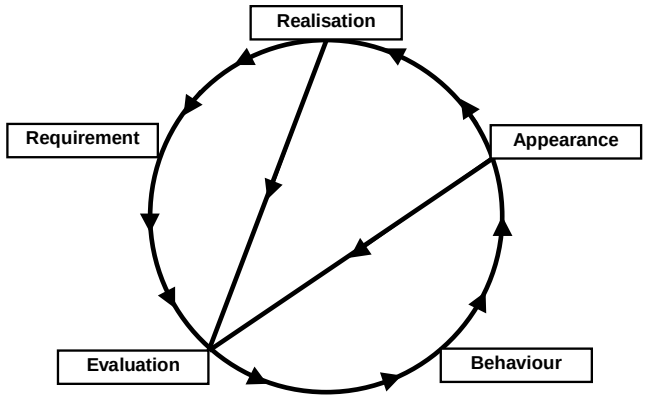


Even if the progression is seen as being recursive (See figure 13 opposite), with the Realisation Point deliverable being used as the input for the Requirement Point, errors will not be picked up until the Requirement Point deliverable has been produced. This increases the potential of incorrect point deliverables being derived from erroneous product of previous development points.

This patently is an oversight unless it is possible to provide access to the Evaluation Point from other points. Such a device to allow this access is partly addressed by the provision for by iteration between sequential points (Characteristic 5). Clearly the recursive model as shown in figure 11 can provide a path for iteration to a previous point by back tracking. This permits the free movement between the Requirements and Evaluation Points and the Evaluation and Behaviour Points encompassing the evaluation process but prohibits the Appearance and Realisation Points of the development process being evaluated.

In order to realise this need for error trapping (Expectation 4) lines of communication between the Appearance and Realisation Points must be provided if successful evaluation of the deliverables for these points can be assessed.

Figure 14: Lines of Concern to the Evaluation Point.



Placing these additional lines on the recursive model provides iterative augmentation as shown in figure 14 above. These new lines of communication can be seen as additional *lines of concern* to the main process of development.

3.2.3 Management Issues.

Each point of the model must provide devices that enable effective management of the process of world building (Objective 3). These devices will encompass reviews and milestones for the development project, allow allocation of resources, provide documentation for the project and support the decision-making processes required of a project manager

One strategy to achieve the successful management of the method and ensure that these requirements are met is to decompose the project into areas of key activities (Characteristic 3). The partitioning of the method for management purposes has effectively already been realised by the five points of the model; Requirements, Evaluation, Behaviour, Appearance and Realisation derived to support the implementation requirements.

In order to ensure that each of the points provide the required level of management support it is necessary to state what processes are to be included at the process point (Expectation 1)

1. Requirement Point.

The goal of the Requirement Point as stated is to derive as much information as possible with regard to the restrictions imposed by the client's expectations of the world and the availability of resources to the project.

In order to realise this goal in a managed way there is a clear need to document these expectations and resources. This requirement can be decomposed into two specific domains, the documentation of the world requirements and the planning of resources to develop the world.

The documentation of the world must include the product of the Requirement Point investigation into the customer requirements. As such this document is expected too clearly specify the world constraints, the component objects and the required behaviour of the world and its objects. Provision should be made in the document to allow and clearly identify additional information to this specification, as the development is progressed (Characteristic 6).

As the method advocates the use of the inline structure of hierarchy (worlds within the world) it is necessary to uniquely identify the original holding world specification document from the component inline object world documentation. The project can therefore be seen as comprising of a controlling *Universe Specification Document* (describing the collection of worlds within the project) linked to a *World Specification Document* for each component world.

This effectively provides a series of "mini" world development projects allowing resource allocation to each to be conducted in a more effective way. In order to manage these mini projects efficiently, however, there must be a controlling mechanism to provide the planning of the development process.

Research into project management processes concludes that the proposed framework is compatible with current project management techniques and it is therefore considered that the planning of the project resources and deliverables will be conducted within such existing structures. Such project planning will be referred to as the *Universal Project Plan* within this paper for identification purposes.

2. Evaluation Point.

The goal of the Evaluation Point, as stated, is to ensure that quantity and quality of a point product is sufficient to satisfy the world requirements identified at the Requirement Point in order to progress the development process.

In order to satisfy these goals the Evaluation Point must include a review of the point deliverables that enable the manager of the project to make a stop/go decision whether or not to proceed with the development. In order to support the decision-making process the review must a present number of questions.

1. Have any prerequisites for the point under evaluation been satisfactorily completed?
2. Is there enough information contained within the Universe and World Specification Documentation to continue the development process?
3. Are there enough resources allocated to the point in the Universal Project Plan to progress?
4. Have the point deliverables been sufficiently realised to permit progression to the next point in the model?

The model therefore provides the following point product evaluations:

1. On Completion of the Requirements Point.

The quality and quantity of the information held in the Universe and World Specification Documentation must be assessed in order to allow progression to the Behaviour Point. The resources allocated to the Behavioural Point development must also be assessed in order to ensure that the level of allocation is correct. This effectively covers the management of the Evaluation Point and the Behaviour Point.

2. On Commencement of the Appearance Point.

The quality and quantity of the information held in the Universe and World Specification Documentation must be assessed in order to allow work to commence. The resources allocated to the Appearance Point development must be also assessed in order to ensure that the level of allocation is correct

3. On Commencement of the Realisation Point.

The quality and quantity of the information held in the Universe and World Specification Documentation must be assessed in order to allow work to commence. The deliverables of the Behaviour and Appearance Points must also be assessed in order to ensure that they are complete and correct to allow the integration process to commence. The resources allocated to the Realisation Point must be reviewed in order to ensure that the level of allocation is correct.

If the results of the review indicate that all requirements have been satisfactorily achieve then progression is possible. If the review indicate that the requirements have not been met then further work must be conducted to satisfy the requirements by returning to the previous point in the model.

3. Behaviour Point.

The goal of the Behaviour Point, as stated, is to realise the optimum behavioural form of a world in order to satisfy the behaviour node requirements identified at the Requirement Point.

To support this goal any solution or change to a behaviour node form during the Behavioural Point development must be recorded in the World Specification Document for the world. Such documentation is essential if the development of the project is to be auditable and used for estimation metrics in subsequent developments.

4. Appearance Point.

The goal of the Appearance Point, as stated, is to realise the optimum combination manipulation and construction of a of a world in order to satisfy the appearance node requirements identified at the Requirement Point

To support this goal any solution or change to the appearance node requirement of a world made during the Appearance Point development must be recorded in the World Specification Document for the corresponding world. Such documentation is essential if the development of the project is to be auditable and used for estimation metrics in subsequent developments.

5. Realisation Point.

The goal of the Realisation Point, as stated, is to realise the optimum combination of the Appearance and Behaviour Point products in order to satisfy the world requirements identified at the Requirement Point.

In order for the successful management of the Realisation Point to be implemented the process of combining the behaviour node and appearance node components into a new master or *holding world* must be reflected in the World Specification Document. If the

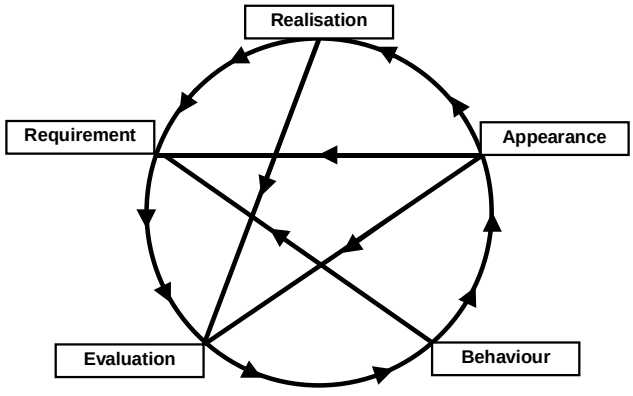
evaluation process for the Realisation point is successful then the Universe Specification Document must also be updated to reflect that the world or world component has been completed. Such documentation is essential if the development of the project is to be auditable and used for estimation metrics in subsequent developments.

Once a world has been "signed off" as complete it can be considered as a object within its own right and can be considered as available, either in whole or its component nodes, for reuse within the current or subsequent projects. An object available for reuse in this manner contributes to a pool of resources or World Resource Library that is available to the development team. Reuse in this manner reduces the potential duplication of effort within a project and increases the efficiency of management planning within the Universal Project Plan.

The need to reflect changes to the Universal and World Specification Documents and the Universal Project Plan as the Evaluation Point assessment may force new requirements to be discovered and to allow the updates from the Behaviour and Appearance Points is made clear from a management perspective. In order to realise these changes the model must be revised to accommodate these additional requirements.

Lines of concern relating to the Evaluation Point process have already been implemented in the model as shown in figure 14 providing all points with an process for evaluation of point product and decision making support as to the progression of

Figure 15: Lines of Concern to the Requirement Point.



development.

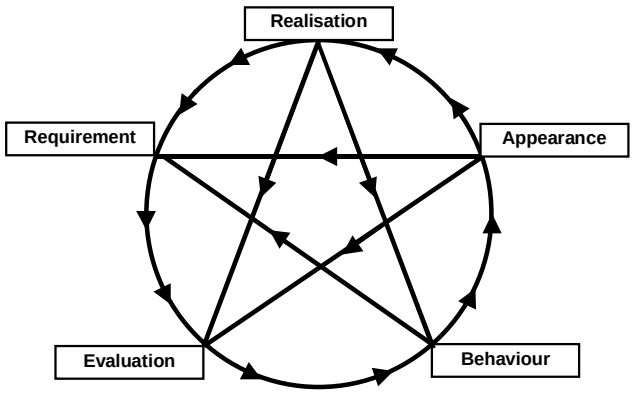
As there are no direct links between the Requirements Point and the Behaviour and Appearance Points, it is necessary to extend new lines of concern if the changes in the Behavioural and Appearance Points are to be reflected within the model, as shown in figure 15.

The addition of these new lines of concern within the model allows the maintenance and development of the Requirements Point product (Characteristic 6) reflecting the need to provide the expectation of changes to the client requirement and specification (Expectation 5). Clearly this extends the scope and potential for error trapping and fault resolution within the method (Expectation 4).

It is noted that the evaluation process for the Realisation Point requires the assessment of both the point deliverables from the Appearance and Behaviour Points in order for assessment to be made. However the possibility those components may have been pulled from the World Resource Library, or that the Behaviour Point product may not have a visual manifestation within the world (such as ambient sound) must be catered for if a deliverable is to be produced (Characteristic 7).

If the method is to permit a reuse strategy and allow for the possibility of disembodied behavioural nodes to provide "visible" output at each discrete process point (Expectation 2) then lines of concern between the Behaviour and Realisation Points must be established.

Figure 16: Line of Concern to the Realisation Point.



Applying this additional line of concern to the model completes the interconnection network of all process as shown in figure 16 (Characteristic 3).

The resulting model permits all points of the model to be passed through in a logical and sequential way by following the circular development path (Objective 2). As each point is passed through the exposure to each of the contributing points is afforded by iteration on the development path for adjacent points and across the lines of concern for nonadjacent points is maximised. This level of exposure permits the refining of requirements and increases the chance early error trapping and resolution whilst still allowing progress to be made on components by reuse, prototyping and the inline hierarchy structure strategies. The overall effect of the development mode is, therefore, a rapid realisation of the world (Expectation 6).

This model represents the simplest possible of interconnection between the five identified development points, the visual form of which is known as a *pentacle*.

3.2.4 Representational Issues.

The development of VRML worlds under the developing method attempts to mirror the way in which humans derive information about the real world by the use of mental models. As such it can be considered as a process of exploring the world possibilities (a functional process) with each point allowing the discovery and refinement of the world and its components (a structural process) Preece (1996).

If the model is to support the methodology there must exist a mapping between structural progression through the model (how it works) and the functional processes or points of the model (how to use it). As part of the previous sections the model's development has been illustrated by diagrams showing the mapping of various lines of concern to the points on the development path this provides the representational form for the structure of the development model (Characteristic 1). Each point process has also been discussed providing a clear statement

of prerequisite, process, sub-process, error path and deliverable thus providing the functional aspect of the model. It is however considered necessary to consolidate these issues here in order to provide a clear understanding of each of the models points (Expectation 1).

1. Requirement Point.

Prerequisite: The need for developing the world either as a new universe or as a component world within a project.

Process: The gathering of information regarding what the client requires and expects the world to do and represent. This process may be achieved by a number of differing methods.

1. Traditional data gathering exercises such as interviews.
2. Schematic representation such as ontological layout diagrams, flowcharting, hierarchy diagrams.
3. The comparison of existing worlds, either form external sources or from the World Resources Library.
5. The development of illustrative prototype worlds and world objects

In addition to this information the Requirement Point allows the investigation into project planning issues. These issues will include; the permitted time scale of the project, target platform, software support tools available, existing VRML and external files held in the World Resource Library and human resource allocation.

Sub Processes: The connectivity provided by the lines of concern to the Behaviour and Appearance Points permits the forward planning of world issues and planning of the individual world object components required.

Such forward planning may result in the development of illustrative prototypes to be used as prompts for further information gathering from the client and development team. While it is considered that prototypes developed in this manner should be discarded, the information gained from their production will augment the World Specification Documentation.

The first draft information for component objects can therefore be utilised as the initial World Specification Document for the component world under the inline structure of hierarchy being refined during the progression along the development path.

Error Path: There is no error path for the Requirement Point as the point itself is concerned with the understanding and communication of the problem domain between all parties involved in the project. If the Requirement Point prerequisite does not exist then the world can never be realised.

Deliverable: The Requirement Point product is a collection of information regarding the worlds to be developed and the resources allocated to the project in order to achieve these. This information exists in three possible documents:

1. The Universe Specification Document, which aims to identify the requirements of the highest level world within the inline structure hierarchy.
2. The Universal Project Plan, which aims to identify all resources available to the project and, provisionally, allocate them to the development points within the project.
3. The World Specification Documents. These documents aim to identify the component objects of the world as individual holding worlds within the inline hierarchy and under the control of the Universe Specification Document.

These documents must exist for each world created in order to provide a definitive audit trail and finalised development documentation.

2. Evaluation Point.

Prerequisite: The Universe Specification Document and the Universal Project Plan. In addition existing World Specification Documents are required for evaluation over the initial circuit, although their presence is optional at the project inception.

Process: The Evaluation Point attempts to establish of the quality and quantity of the reviewed documents in order to make an informed decision as to whether the world under consideration can be advanced to the development stage.

Sub Processes: The line of concern to the Appearance Point permits the assessment of the documentation to establish the progression and possible augmentation of the required appearance node specification for the world.

The line of concern to the Realisation Point permits the assessment of the documentation to establish that the world can be realised according to the Universal and World Specification Documentation. This is conducted by reviewing the documentation for the behaviour node, which will indicate either a static or dynamic behavioural component. The line of concern to the Realisation Point also permits the reviewer to consider the possibilities of further decomposing the world under evaluation into a series of smaller inline worlds within the structural hierarchy.

Error Path: If the Evaluation Point review establishes that there is not enough information to advance to the development point then the project must be backtracked to the Requirement Point in order to further clarify the information under review. Where such backtracking occurs the respective documentation must be annotated as such in order to provide an accessible audit trail.

Deliverable: Refined Universe and World Specification Documentation (behaviour and appearance nodes). It is also possible that the review process will identify a mismatch within the allocation of resources to the development points thereby also requiring the updating of the Universal Project Plan to insure the correct level of resources are deployed to accommodate such shortfalls. All deliverables successfully passing the Evaluation Point reviews must be signed of by required personnel (such as project manager, development team leader, client etc) in order to provide an accessible audit trail.

3. Behaviour Point.

Pre Requisite: The Universe and World Specification Documents and the Universal Project Plan. In addition if a behaviour node is to be developed for an existing world from the World Resource Library the VRML world file or script for that object. If no behaviour is associated with the world the development may be progressed to the Appearance Point.

Process: The Behaviour Point aims to allow the creation and exploration of the behavioural requirement for the world as detailed in the World Specification Document and compliant to the Universe Specification Documentation. This behaviour may be implemented as dynamic, static or by combination of these two forms.

Any new functional requirements or solutions should be documented within the World Specification Document even if they are not implemented in order to provide an alternative implementation route should the Realisation Point process fail.

Sub Processes: The line of concern to the Requirement Point permits the investigation of the desired behaviour either as a complete behavioural node or as a series of inline behaviour nodes based on the information held in the World Specification Document.

Such investigation may be trialed with either prototype appearance nodes or with the existing world from the World Resource Library by virtue of the line of concern to the Realisation Point. Such advance realisation of the behavioural node may establish the behaviour as being generic and a possible candidate for inclusion into the World Resource Library.

Error Path: If the investigation and creation of the behavioural node establishes that the desired behaviour cannot be implemented a process of backtracking to the Evaluation Point must be conducted. This is in order to establish whether the behavioural information as held within the World Specification Document is incorrect or missing or whether there has been a misallocation of resources. The project documentation must be annotated to reflect this backtrack in order to provide an accessible audit trail.

Deliverables: The behavioural node script for the VRML world under consideration. This script may have a companion appearance node depending on the use of prototypes of existing components from the World Resource Library. The project documentation must be signed off by the required personnel (such as the project manager, development team leader, client etc.) in order to provide an accessible audit trail.

In addition if the behaviour node is to be considered for inclusion into the World Resource Library it must be submitted to the World Resource Librarian for further evaluation and documentation to establish compliance with the reuse policies in effect for the organisation. The World Resource Librarian is a member of staff responsible for the husbandry and maintenance of the World Resource Library files. This may be a nominal role within the team for small-scale VRML developments or a dedicated member of staff for larger projects.

4. Appearance Point.

Prerequisite: The Universe and World Specification Documents and the Universal Project Plan. If no visual form is associated with the world the development may be progressed to the Realisation Point.

Process: The Appearance Point aims to permit the creation and exploration of the visual form of the world as detailed in the World Specification Document and compliant to the Universe Specification Documentation.

Any new visual requirements or solutions should be documented within the World Specification Document even if they are not implemented in order to provide an alternative implementation route should the Realisation Point process fail.

Sub Processes: The line of concern to the Requirement Point permits the investigation of the desired visual form either as a complete appearance node or as a series of inline appearance nodes based on the information held in the World Specification Document.

The line of concern to the Evaluation Point allows the informed decision to progress the world on to the Realisation Point to be made. As the prime medium of VRML is by its nature graphical for either of the available specifications this review is clearly of paramount importance if the quality of the world is to be maintained.

Appearance nodes that successfully achieve this second evaluation may be considered as possible candidates for inclusion into the World Resource Library.

Error Path: If the investigation and creation of the appearance node establishes that the desired visual image cannot be implemented a process of backtracking to the Evaluation Point via the line of concern must be conducted. This is in order to establish whether the appearance information as held within the World Specification Document

is incorrect or missing or whether there has been a misallocation of resources. The project documentation must be annotated to reflect this backtrack in order to provide an accessible audit trail.

Deliverables: The appearance node script for the VRML world under consideration. The project documentation must be signed off by the required personnel (such as the project manager, development team leader, client etc.) in order to provide an accessible audit trail.

5. Realisation Point.

Pre Requisite: The Universe and World Specification Documents and the Universal Project Plan. In addition to these documents the VRML files or scripts that comprise the world appearance and behavioural nodes. As previously noted the holding world under consideration at the Realisation Point may have a behaviour world node, appearance world node or a combination of both. It is further noted that these world nodes might themselves be constructed of subordinate holding worlds within the inline hierarchy structure.

Processes: The Realisation Point aims to successfully assimilate the world appearance and behaviour nodes in order to permit the creation and exploration of the holding world as detailed in the World Specification Document and compliant to the Universe Specification Documentation. This process can be viewed as two distinct operations

1. The internal assimilation of the holding world by the of its inline component nodes, subsequent global manipulation of those components and testing to ensure compliance to the World Specification Document.
2. The external integration of the holding world as an inline component of another world higher in the hierarchy of the inline structure, subsequent global

manipulation of the world as a component item and testing to ensure compliance to the higher order World Specification Document or Universe Specification Document.

Any new world requirements, solutions or interaction issues discovered by the assimilation process should be documented within the World Specification Document or Universe Specification Document, respectively, in order to provide an accessible audit trail and provide decision support information should the Realisation Point process fail.

Sub Processes: The line of concern to the Appearance point permits the possible refinement of the appearance of the holding world within its host world by use of level of detail, delayed loading etc based on the information held in the World Specification Document or Universe Specification Document, respectively.

The decision as whether to allow the holding world to be included in the final universe, or to continue development of the world by progression to the Requirements point for rework is afforded by the line of concern to the Evaluation Point. Clearly this additional review must be held in order to maintain the quality of the world.

Error Path: If the assimilation process conducted at the Realisation Point establishes that the holding world cannot be implemented according to the information presented in the host World Specification Document or the Universal Specification Document, respectively a number of choices are presented:

1. If the world under consideration can be partially realised by any of its subordinate world nodes then the holding world may be considered as an exploratory prototype. Such prototypes may be recursively passed on to the Requirement Point to be used to establish further information. It is suggested

that the World Specification for a holding world progressed as a prototype be signed off and removed from the mainstream documentation to an archive file in order that the information contained within the document is not “resurrected” accidentally. The project documentation must be annotated to reflect this progression as a prototype in order to close the world audit trail.

2. If the appearance node of the holding world cannot be realised then a process of backtracking to Appearance Point may be considered in order to attempt to rework the appearance node. This process should take into account any additional alternative appearance node implementations as documented within the World Specification. The project documentation must be annotated to reflect this backtrack in order to provide an accessible audit trail

3. If the behaviour node of the holding world cannot be realised then a process of backtracking to Behaviour Point (as provided by the connecting line of concern) may be considered in order to attempt to rework the behaviour node. This process should take into account any additional alternative behaviour node implementations as documented within the World Specification. The project documentation must be annotated to reflect this backtrack in order to provide an accessible audit trail

4. If none of these options are taken or are unavailable then backtracking to the Evaluation Point must be conducted (as provided by the connecting line of concern). This must be done in order to establish whether information held within the World Specification Document is incorrect or missing or whether there has been a misallocation of resources. The project documentation must be annotated to reflect this backtrack in order to provide an accessible audit trail.

Deliverables: The assimilated and integrated VRML holding world file as detailed in the World Specification Document or Universe Specification Document, respectively. The project documentation must be signed off by the required personnel (such as the

project manager, development team leader, client etc.) in order to provide close to the world audit trail.

A holding world that has successfully completed the Realisation point may now be considered for inclusion in the World Resource Library for reuse. The world and a copy of the development documentation should therefore be submitted to the World Resource Librarian for consideration as a resource world within the Library.

It is considered that this statement of the expected prerequisites, processes, sub-processes, error path and deliverables successfully provides a representation of the structural and functional processes of the method. It is noted that such a statement of expectation caters to all parties involved within the project, the client, the project management team, the quality assurance personnel, the behaviour engineers and the aesthetic designers by identifying their roles within the differing points of the model (Expectation 3).

In order to clearly identify the different development components of a world it is considered necessary to provide some form of graphical notation that is easily understood by both the development team and the client (Characteristic 2). This is imperative if the Universe and World Specification Documents are to reflect the inline hierarchy used within the project.

Previous sections have introduced a number of new terms in an attempt to describe which world is being referred to at any one point. If such a graphical notation is to be developed to detail the construction of the inline hierarchy then a set of symbols must be introduced to accommodate these new terms (Expectation 1).

It is therefore considered valuable to review these terms as follows.

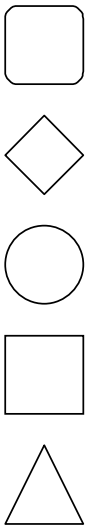
1. The Universe: The top-level world of the inline structure hierarchy within which all other worlds reside.
2. A Holding World: An object within the Universe that may be represented by other inline worlds constituting the appearance and or behaviour of the object.
3. The Appearance Node (World): A world that describes the visual form of an object.
4. The Behavioural Node (World): A world that describes the behavioural form of an object.

In addition to these terms another must be introduced, the external files type. External files are those which are not either VRML or behavioural script files but are either used by or accessed from a world. As such these would include, image files used for texture mapping, sound files, links to HTML pages and applications called via behavioural scripts.

Research for the previous chapter indicated that one of the criticisms levelled at current software engineering paradigms was the lack of an understandable support methodology. It is suggested that the key player in the understandability issues is the ability to easily communicate the concept to all involved parties by a set of clear, representative and iconic notation (Expectation 3).

It is therefore proposed that the graphical notation consist of simple primitive shapes or icons arranged in an order that accurately reflects the inline hierarchy as shown in figure 17 opposite.

Figure 17: Proposed Iconic Representation.



The Universe Icon.

The Holding World Icon.

The Appearance Node Icon.

The Behaviour Node Icon.

The External File Icon.

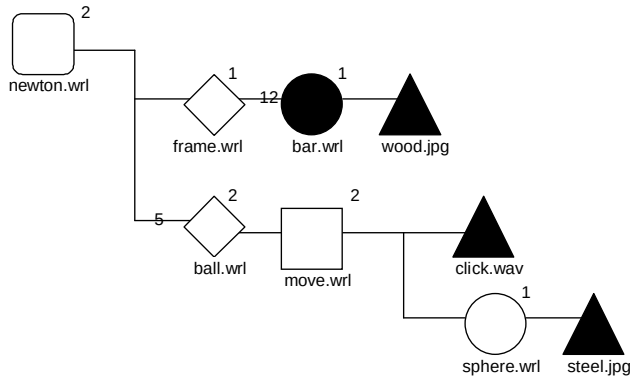
Each of these icons may be further augmented with the inclusion of the world URL or file name and a superscript numeric indicating the VRML specification for the file. Connectivity between these files is to be shown be the use of unbroken straight or angled lines. Where multiple instances of a inline world displaying similar base appearance and behaviour forms occurs within the same world, the addition of a numeric on the connecting line is permissible in order to conserve space.

It is suggested that until a world has reached the Realisation Point within the development model that the icon remains blank, and that upon completion and validation the icon is filled in, indicting completion and submission to the World Resource Library.

It is noted that there may exist a situation where a world may purely consist of one of the two node world types. When such a world exists it is considered legitimate to dispense with the parent world icon (the diamond) if no direct processing or manipulation of that world is required.

Thus a VRML world representing a simple Newton's Cradle with five balls under the given notation would appear as shown in figure 18 opposite. The figure shows that the file NEWTON.WRL is of the VRML 2 file specification and consists of two subordinate worlds. FRAME.WRL is a VRML 1 world consisting of 12 instances of BAR.WRL each utilising an external flat format graphics file WOOD.JPG.

Figure 18: Example Hierarchy Plan for "Newton.wrl".



The other component objects of the world are 5 instances of the VRML 2 world BALL.WRL, which have a behavioural node, MOVE.WRL utilising the external file sound file CLICK.WAV. The MOVE.WRL also utilises a VRML 1 file SPHERE.WRL which calls upon the external flat format graphics file STEEL.JPG. The figure also identifies the file BAR.WRL and the external files as having been completed or is ready for use.

3.2.5 Learning Issues.

The previous three sections have detailed the concerns of the implementation, management and representation of the pentacle model and its associated methodology in relative isolation from each other. While these sections have shown the logical development of the model from first principles it is considered that the information is not readily access able to the potential user of the methodology.

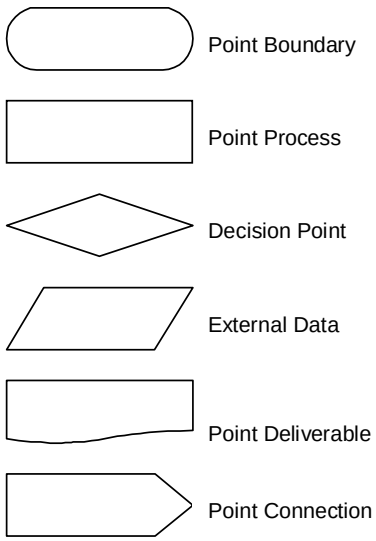
If the methodology is to be utilised then there must exist a synthesis of the presented information in order to permit ease of learning for those new to the paradigm (Characteristic 1).

Such a synthesis must detail the boundaries of each of the five identified development points (Expectation 1) in a manner that is understandable to those parties involved within the development project (Expectation 3). This synthesis, or *manual* must also detail when events must and may take place and what must be recorded in order to maintain the quality issues of the project.

From the previous sections it is clear that the manual should rely on graphical notation to impart such knowledge (Characteristic 2) and that such a notation must be both comprehensive and familiar (Expectations 1 and 3). The most logical candidate for the manual notation would be the reuse of the iconography already proposed. However it is considered that the reuse of the symbols specific to VRML world planning will ultimately lead to confusion and misinterpretation by the method user (as noted in the previous chapters this is a common criticism of existing software engineering paradigms).

In order to reduce potential confusion between an additional new notation and the proposed iconic representation outlined above for VRML world planning, a number of existing notations for indicating data flow were reviewed. It is suggested that the internal process flow of a point should utilise the generic flow chart devices as shown in figure 19 opposite.

Figure 19: Reused Flow Chart Representation.



While the mixing of notations may seem strange the use of the existing notation devices is considered legitimate for a number of reasons;

1. It is considered that the notation clearly indicates the possible sequence of processing within each of the points.

2. The notation is well established and therefore will be recognisable to a larger group of potential users.

3. The iconography of the notation differs sufficiently enough from that proposed for the planning of VRML worlds to avoid confusion.

4. The notation is not required for reproduction during the development process as it is purely an aid to learning. Therefore the complexity of some of the icon devices is not a restriction to the development process.

It is considered, therefore, that the issues of learnability surrounding the new model and its associated methodology can be successfully addressed by the combination of the model its self, the methodology's native notation and supplemented by flow chart devices to indicate process flow within each development point.

3.3 Conclusion of New Ideas Considerations.

It is considered that the issues surrounding the development of a new method that directly supports the processes required to build quality VRML worlds have been logically drawn from first principles and discussed in order to present a frame work to build such a method on.

In accordance to the requirements of the project and the need for formalising the issues discussed as identified the Pentacle Model and its associated methods must be synthesised into some form of cohesive manual. This assimilation of the discussed issues will therefore constitute the development chapter of the paper that follows.

4 Method Development.

"A good manager can manage any project … if he or she is willing to learn the milestones that can be used to measure progress, apply effective methods of control, disregard mythology, and become conversant in a rapidly changing technology"

Roger Pressman (1994)

The previous chapter has shown the logical and progressive development of the Pentacle model and its associated methodology. It has been noted that during this process that a number of different view points of the model have been presented and that in order to successfully realise the methodology as a usable paradigm a manual must be produced from these different strands.

From those concerns raised by the learning issues it is proposed that the manual consist of a textual explanation of the model point processes illustrated by flow charts indicating the sequence of order. Having discussed the principles of methodologies in general and considered the salient features of a new paradigm to support the building of quality VRML worlds (the Pentacle Model or more correctly the *Pentacle Method*) in previous sections it is considered that the manual be produced commencing with an overview of the method.

It should be noted that it is assumed that the paradigm will be utilised by world builders who have a prior knowledge of the fundamental syntax, structure and concepts expressed within the VRML language. It is therefore considered that references to VRML specific implementation techniques and terms do not need to be explained within the narrative of the manual.

4.1 The Pentacle Method: An Overview.

The Pentacle Method provides a paradigm for the construction of quality VRML worlds that enables the process of world building to be conducted in a logical and manageable sequence. The method utilises a model that is divided into five delimited development points each connected to each other to form a simple network representative of its development process.

These points characterise the processes that must occur at a given phase of the world building project, and are identified, in respective order, as follows.

1. Requirement Point: The processes establishing the project scope and available resources.

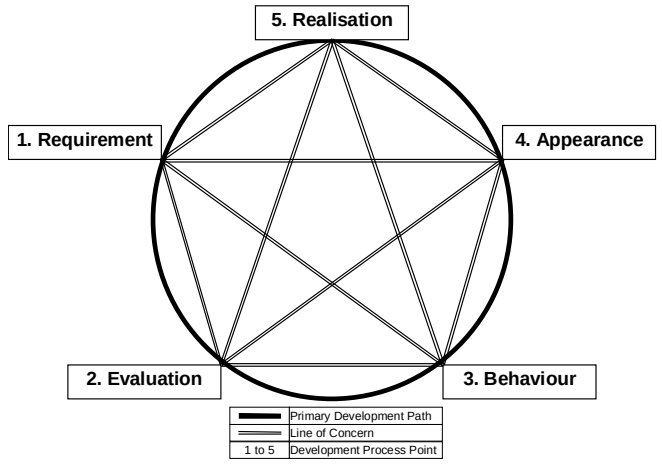
2. Evaluation Point: The processes that control the decisive quality issues and advancement of the development.

3. Behaviour Point: The processes utilised in the construction of a world's behavioural component.

4. Appearance Point: The processes utilised in the construction of a world's appearance component.

5. Realisation Point: The process governing the physical structure, optimisation and assimilation of a world's component parts.

Figure 20: The Pentacle Development Model.



The model provides a recursive regime permitting the progression from the Realisation Point to the Requirement point after the initial circuit to allow refinement of development. Each point is considered as having two primary development paths and two secondary development paths or lines of concern. Iteration between points is either provided directly by the primary development paths by backtracking to the previous point or by cutting across the primary development path to another point by the lines of concern. The Pentacle model is illustrated in figure 20 above.

The process of world building is conducted by following the primary development path for each of the five points in a recursive sequence. Although each point may break the primary development path by virtue of its line of concern development conducted by such a deviation from the primary development path is considered exploratory and requires that the primary development path be return to on the completion of the diverted point processes. Such point exploratory or prototype development is made available for review and refinement when the primary development path next accesses the point.

Development recursively continues along the primary as a process of discovery, exploration refinement and completion until all the requirements of the project have been addressed. There is, therefore, no limit to the number of circuits that can be made during the duration of the project.

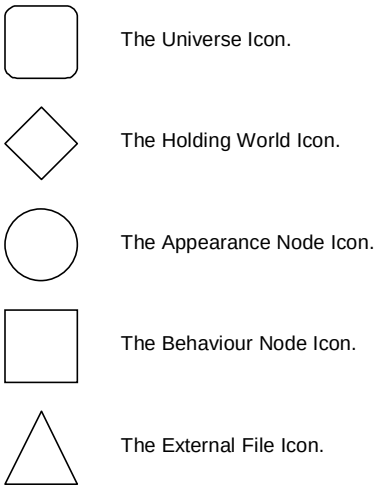
In order to simplify the world building process and reduce the size of the generated VRML file the Pentacle Method advocate the use of the VRML inline technique for constructing world scenes. Each world inline component is therefore treated as worlds within their own right accessed by a master world or *universe*. Each component or *holding world* is seen as having two possible components, behaviour and appearance. These two types of component worlds are identified as *node worlds* in order to distinguish them from normal holding worlds.

The process of decomposing the universe world into a series of inline holding worlds and node worlds provides a top down analysis of the universe requirements. These requirements are then utilised to build the universe from the bottom up by the assimilation of node worlds and holding worlds within the inline hierarchy.

In order to plan and track these inline worlds the method utilises a simple hierarchy notation. The notation consists of five icons, three represent three different kinds of world files, holding, appearance and behaviour, with the remaining two indicating the universe world and external files that may be accessed or utilised by the worlds.

The icons used by the notation are shown in figure 21 opposite.

Figure 21: The Inline Notation.



Each of these icons are subscripted by the name for the file which the icon represents and is superscripted with a numeric indicating which version of VRML the file format is compliant to (1, 1.1, 2 etc). External file icons clearly do not need such identification of VRML compliance and should not be annotated.

Starting with the universe icon the component holding worlds are layered in a tree structure reading left to right (for landscape paper orientation or top to bottom if using portrait paper orientation). Solid lines are used to indicate inclusion to the respective holding world or the universe world icon by connecting various the file icons. Where multiple instances of a single inline world occur within a holding world (or the universe world) the connecting line may be annotated with the appropriate number of instances that occur in order to save space.

On completion of the development of a world (or acquisition of external file) the icon is filled in to indicate that the world is available for use. Completed files should be considered for inclusion into a centralised database of developed objects, the World Resource Library.

The submission of worlds to the World Resource Library is a process of the method requiring the world under submission to undergo a quality assurance program involving. The paradigm does not detail the processes that must occur for this quality assurance process this being left to the individual quality assurance staff or librarian. It is however recommended that consideration be given to factors such as;

1. Compliance checks against the appropriate VRML specification.

2. The optimisation of the world by elimination of redundant or unnecessary code.

3. Checks to insure high cohesion of the world.

4. Checks to insure loose connectivity with other worlds.

5. The refinement of the World Specification Documentation enabling reuse of the world.

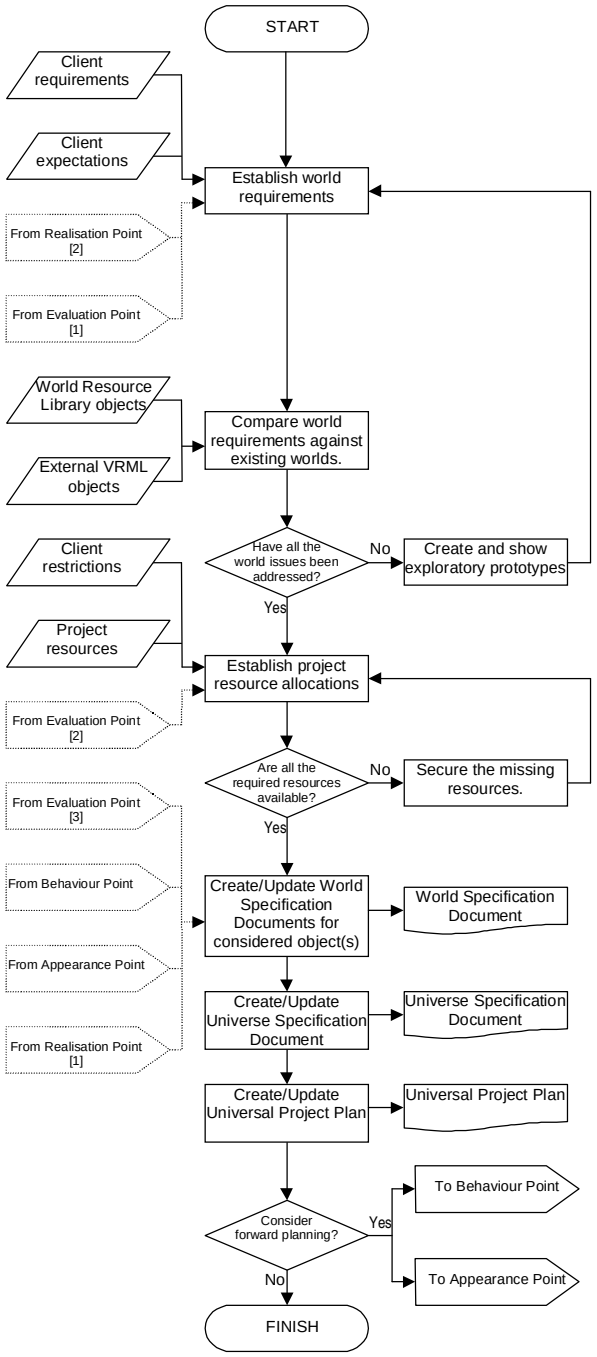
6. Conversions to the latest VRML specification to ensure the techniques employed have not been superseded.

As the project continues (and as more projects occur) the World Resource Library will grow providing the developer with pre-built solutions for common worlds and behavioural scripts ready for reuse.

4.2 The Pentacle Method: The Requirement Point Process.

The Requirement Point of the Pentacle Method is concerned with establishing the client's requirements for the world to be developed. The Requirement Point also provides processes for the planning of both the project development and the allocation of available resources to the project required to realise that world.

Figure 22: The Requirement Point Process.



The initial process of the point attempts to extract as much information as possible from the client with regard to the world requirements and the expectations the client has of the medium. While the necessity to establish the requirements of the world is obvious the need to derive the clients expectations of VRML is perhaps not so clear. This process is necessary in order to correct any misconceptions the client may have acquired from the degree of hyperbole surrounding the abilities of VRML and 3D graphics in general. By such an exploration of client expectations erroneous preconceptions can be dispelled thus reducing the chance of commencing development of a world that is beyond the scope of the media

In addition to these two requirements additional information provided by previous circuits of the development path may be available. These are information already gathered but considered incomplete or inaccurate by the Evaluation Point processes and prototype worlds developed thus far from the Realisation Point processes. Clearly these two information sources will be unavailable in the initial circuit of the primary development path.

Once the client's requirements and expectations for the world to be developed have been established a search for similar worlds from the World Resource Library and external sources is conducted. The object search has three purposes

1. It allows comparison with existing worlds to enable the client to see what is possible to achieve with the media.
2. It may prompt the client to express requirements not previously considered or identified.
3. It permits the identification of reusable component worlds at an early stage thus removing them from the development process.

Having completed the first two processes of the Requirement Point a review of the information gathered thus far must be held. This review point is primarily conducted to ascertain whether all of the issues surrounding the client view of the world have been addressed.

If it is considered that there are areas that require consolidation or further clarification illustrative prototypes may be developed and demonstrated to the client. This use of prototypes provides a loop back to the initial two processes of establishing requirements and existing world comparison in order to address the missing information. Prototypes developed for illustrative purposes by their nature are rough implementations of specific world components and are therefore considered to be disposable.

If it is considered that all of the issues surrounding the world development have been successfully addressed then the project may now have resources allocated to it.

Initially there are two factors that must be considered when establishing the allocation of resources to the world development project, client restrictions and available resources. Client restrictions may manifest themselves in a number of forms, time scale, target platform, preferred browser client held data and so forth. Resources available to the project that must be considered are availability of software tools, range of platforms and browsers, and staffing levels.

Supplemental to these two factors there may also be an additional resource requirement identified during the Evaluation Point process from previous circuits of the primary development path. Clearly this additional information will not be available for the initial circuit of development.

On completion of the resource allocation point a review must be held in order to establish if any resources are under subscribed or missing. If such a resource is thus identified, the opportunity is presented to acquire the resource in order to redress the correct resource levels according to the client restrictions and world development requirements.

A process of documenting the requirements and resources allocated to the project now commences. This documentation is comprised of three types of dynamic documents, World Specification Documents, the Universe Specification Document and the Universal Project Plan. The initial circuit of the primary development path will cause the creation of these documents with subsequent circuits providing additional information and refinements as the project continues. In addition subsequent circuits may produce new World Specification Documents as worlds are decomposed into component parts.

A World Specification Document is required for each component world that is to be developed within the scope of the project. There are three types of worlds that a World Specification Document may describe;

1. Appearance Node Worlds:

A world dealing specifically with the appearance of an object. Generally referred to as an appearance node.

2. Behaviour Node Worlds:

A world dealing specifically with the appearance of an object. Generally referred to as a behaviour node

3. Holding Node Worlds:

A world comprising of one or more node worlds of any type. Generally referred to simply as the world under consideration or the world.

The World Specification Document should detail the type, purpose and position within the inline hierarchy for the world in question. Subordinate objects to the world in question should be identified by use of an Inline Hierarchy Diagram.

The world described by the Universe Specification Document is designated as the highest holding node world within the world inline hierarchy. This world contains all call subordinate inline worlds as detailed in their individual World Specification Documents and global considerations such as lighting, camera view points ambient sound and so forth. Unlike World Specification Documents there will only be one Universe Specification Document for a project.

The Universal Project Plan is a generic term used within the Pentacle method to describe any project management planning device within the project domain. Such devices will depend on the management systems employed with the project but should include projections of project time scales, staff utilisation, resource deployment and so forth. The information needed to

compile the Universal Project Plan is directly connected to the project resource allocation process but may be refined by subsequent circuits of the primary development path.

In order to capitalise on the momentum generated by the documentation processes the opportunity is afforded to pre-empt the client approval and commence prototype development of the project world and its components. This process is referred to as forward planning and requires the advance commitment of resources for the Behaviour and Appearance Point processes.

Forward planning carries a proportional risk consideration with it as the client may not approve the project as planned, or cancel the development totally. However this must be weighed against the unused time available between client project document submission and approval which may be utilised to produce first draught evolutionary prototypes, experiment with ideas or consolidate additional soft resources.

The conclusion of the Requirement Point requires the termination of any development instigated by lines of concern to be halted at this point and normal development is returned to the primary development path. It is at this stage that all of the relevant information pertaining to the project thus far gathered must be presented to the client for acceptance. In the case of post initial circuits it may not be necessary to contact the client if the particular development under consideration has already been approved.

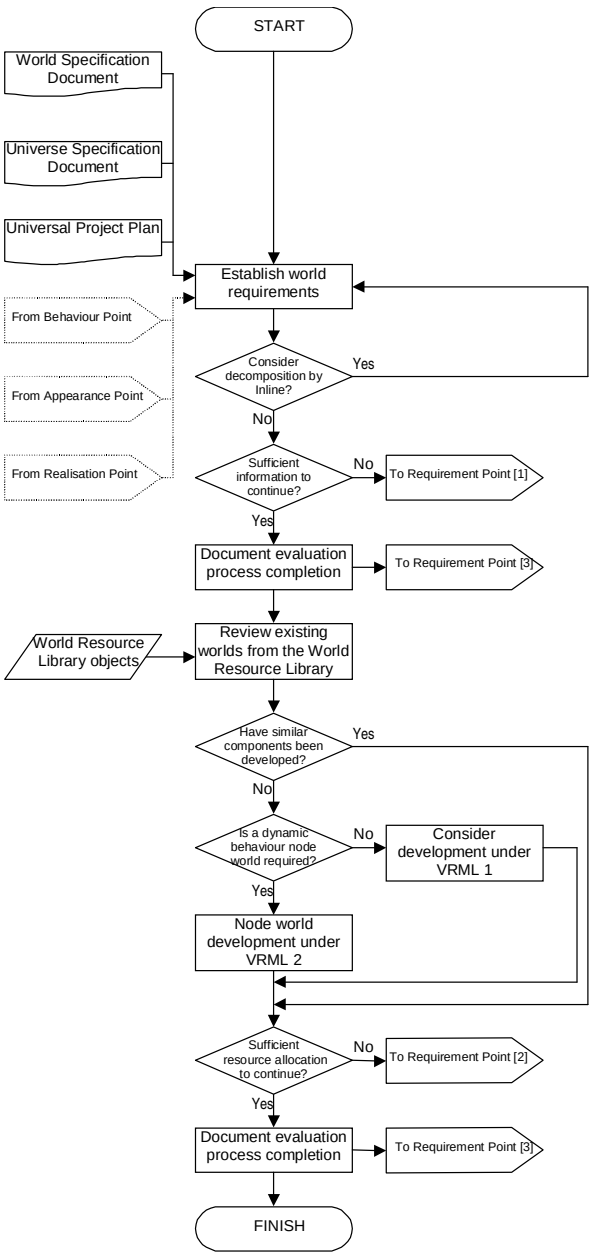
4.3 The Pentacle Method: The Evaluation Point Process.

The Evaluation Point of the Pentacle Method is concerned with evaluating the information gathered from the Requirement Point and refinements identified by other process points in order to ensure that development may continue. The Evaluation Point also provides processes to establish initial inline decomposition, world reuse and VRML specification to be used for the realisation of the world under consideration.

Access to the Evaluation Point by the primary development path is only possible once the deliverables for the preceding Requirement Point have been agreed.

The initial process of the point requires a review of the information contained within the World Specification Document for the world under consideration. As evaluation of the information for the world under consideration may impact on the project in the broader view it is also recommended that a review of the Universe Specification Document and Universal Project plan also be held to establish those issues directly connected to the world under consideration.

Figure 23: The Evaluation Point Process.



In addition to the project documentation information derived from previous circuits of the primary development path may be available. Clearly such additional information will not be available on the initial circuit of the primary development path.

Having established the world expectations and requirements the possibility of decomposing the world under consideration into a series of inline worlds under a holding world should be considered.

If inline decomposition is considered possible then the previous process of establishing the new holding and inline world requirements must be revisited for each of the new worlds considered.

If no further decomposition is possible or required at this stage then the quality and quantity of information from the initial requirement review process must be assessed as to its relevance and completeness. Where such information is missing, erroneous, vague or conflicts with the existing documentation the Requirement Point must be revisited in order to rectify the identified problem.

Having completed the world requirements review process any changes to or creation of any new World Specification Documents must be recorded by temporarily breaking the primary development path and returning to the Requirement Point.

Using the refined World Specification Documentation the objects available in the World Resource Library should be reviewed in order to ascertain if the requirements of the world match any of the existing developed worlds. If such worlds have already been developed then these may be reused, if not then the behavioural requirements must be assessed in order to ascertain the version of VRML to be used to realise the desired behaviour.

The World Specification Document for the world under consideration should have identified the required behaviour type for the world. If dynamic behaviour is required then clearly the VRML 2 specification should be used as the development platform. Where the world requires either static behaviour or has no behavioural requirement then the use of VRML 1 specification may be considered.

This potential for splitting the development platform enables the planning of human resources to be more effective by capitalising on the skill sets available within the development team. It is assumed that more world builders will be familiar with the older VRML 1 specification than the newer version 2. It is noted however that as the language is developed this familiarity will shift towards the newer specification.

Once the development version and reuse issues have been established the project documentation for the world under consideration should be reviewed once again to insure that the resource allocation is sufficient to allow the continuation of the development. If the resource allocation is found to be inadequate then the Requirement Point must be revisited to re-establish the balance of resources allocated to the different developments within the project.

The final process of the Evaluation Point requires that any refinements, discoveries, platform and reuse issues be recorded in the appropriate project documentation.

The completion of the Evaluation Point process should have established and refined the project documentation in regard to the world under consideration and resolved any potential hindrances to further progression on the primary development path to the Behaviour Point. The Evaluation Point process finalisation requires that processes instigated by lines of concern to the Evaluation Point are halted and normal development returned to the primary development path for the calling point process.

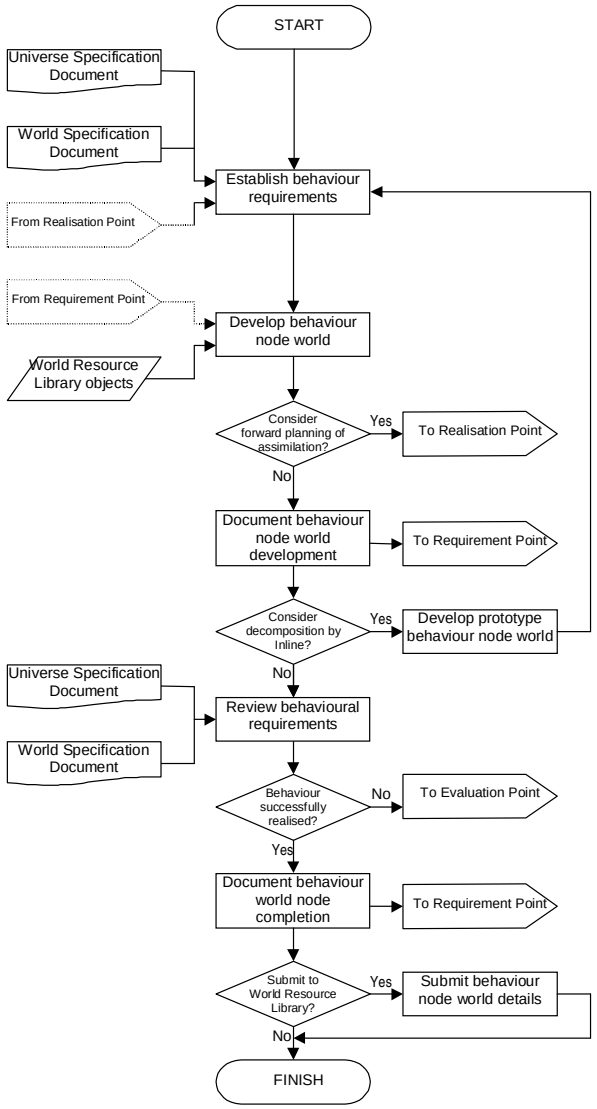
4.4 The Pentacle Method: The Behaviour Point Process.

The Behaviour Point of the Pentacle Method is concerned with constructing behaviour for the world under consideration according to the information held within the World Specification Document. The Behaviour Point also provides processes to further establish inline decomposition and experiment with holding world prototypes.

Access to the Behaviour Point by the primary development path is only possible once the Evaluation Point processes have been completed. If no behavioural development is required for the holding world then the holding world documentation should be marked as such curtailing the Behaviour Point development process and progressing the primary development path on to the Appearance Point.

The initial process of the Behaviour Point requires a review of the World Specification Document for the world under consideration and the role of the world within inline hierarchy as presented in the Universe Specification Document.

Figure 24: The Behaviour Point Process.



In addition to this information additional data may be available from the Realisation Point for the world after the initial circuit.

Once the behavioural requirements of the world under consideration have been clearly established the process of developing the world behaviour can commence. Such development may be conducted in order to generate a new behaviour node world or may make use of existing behavioural objects from the World Resource Library as identified in the World Specification Document for the world under consideration.

The VRML recommended platforms for developing world behaviour are native VRML nodes for static behaviour and Java Script for dynamic behaviour. It should be noted that alternative platforms are available for realising world behaviour.

It is at this point that worlds considered for forward planning at the Requirements Point enter the behaviour development process. Worlds entering the Behaviour Point development process by this means should be considered as prototype worlds until ratified by the Evaluation Point processes.

Once the behaviour node has been developed the possibility for forward planning with regard to assimilating the node within it immediate inline hierarchy may be considered. Such forward planning of assimilation can only be considered if the behaviour node holding world's World Specification Document indicates that all subordinate node worlds and external files are available for assimilation. This option will generally only be taken where such holding world components have already been developed or are available from the World Resource Library.

If forward planning for the behavioural node assimilation is required the primary development path is temporarily broken and development is moved to the Realisation Point. As with forward planning within the Requirement Point a world taking the forward planning route is considered

as a prototype assimilation that will be refined when the primary development path enters the Realisation Point.

Having produced the behaviour node and possibly explored assimilation the development must be recorded in the appropriate project documents. The documentation should include the behavioural solution implemented and any considered but discarded approaches to the problem domain. This is required should the derived solution fail during the final assimilation at the Realisation Point and can be utilised as a starting point for developing corrective alternative solutions.

Using the documented solution the derived behaviour node should be examined in order to ascertain if it can be further decomposed by the use of the inline hierarchy. If such decomposition is possible then the inline subordinate behaviour nodes must be prototyped and the initial review point returned to establishing the behaviour requirements of the new node worlds.

On the completion of the behavioural development loop a final review of the refined World Specification Document for the world under consideration and its role within the inline hierarchy as indicated by the Universe Specification Document must be held. The purpose of this review is to establish that all of the requirements detailed in the World Specification Document have been addressed to an appropriate level. If the review identifies that the behaviour does not meet the required criteria as specified then the unsuccessful implementation may be attributed to a number of factors;

1. The allocation of resources to the node development may have been inappropriate or under subscribed.
2. The information contained within the World Specification Document for the node may be erroneous, misleading or vague.

3. The behaviour as required by the World Specification Document may be unrealisable and alternative solutions may need to be sought.

A world failing this review point may be considered as an exploratory prototype world but the Evaluation Point must be returned to so that the World Specification document and resource allocation may be re-evaluated. Any subsequent development for the world must be halted and the re-evaluated world is treated as a new development thereby allowing the Behaviour Point to be re-entered at the starting point.

Those worlds passing the review point may now be considered as being complete and should be documented as such within the World Specification Document, the Universe Specification Document and the allocated resources within the Universal Project Plan be returned to the project resource pool for reallocation.

The complete behaviour node world may now be considered for inclusion into the World Resource Library. In order to maintain the quality content of the World Resource Library a candidate object for submission should fit into one of the following categories.

1. The developed behaviour represents a “standard” behaviour node that can be widely reused.
2. The developed behaviour performs one simple task that may be combined with others to make a more complex node world. In this case the behaviour can be seen as a behavioural primitive like the optimised primitive shapes offered by VRML.
3. The developed behaviour is a highly complex node that would take much time and resources to recreate.
4. The developed behaviour offers an alternative behavioural solution to an existing node world within the World Resource Library.

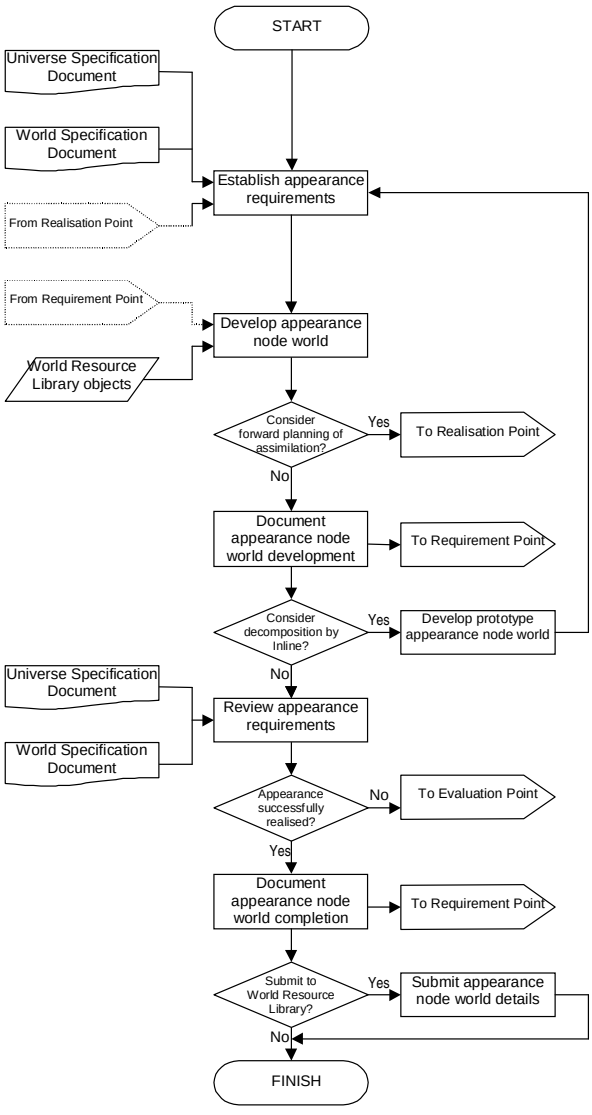
If the option for inclusion is taken then a copy the behaviour node world and its associated World Specification Document must be passed to the librarian for optimisation and clarification of documentation. A world submitted to the World Resource Library may continue along the development process until the Realisation Point where the copy world may be substituted for the purpose of assimilation if the optimising process has been completed.

The conclusion of the Behaviour Point development process requires that all processes initialised by the lines of concern from external points now be terminated and the primary development path be restored. The Behaviour Point will have produced a stable behaviour node world in accordance to the World Specification Document that may be used in demonstrations to the client if required. The development point must be signed of as complete before progression is possible to the Appearance Point development processes.

4.5 The Pentacle Method: The Appearance Point Process.

The Appearance Point of the Pentacle Method is concerned with constructing graphical "look and feel" of the world under consideration according to the information held within the World Specification Document. The Appearance Point also provides processes to further establish inline decomposition and experiment with holding world prototypes.

Figure 25: The Appearance Point Process.



Access to the Appearance Point by the primary development path is only possible once the Behavioural Point development process has been completed (for holding worlds without behaviour this may simply be signed off). If no appearance development is required for the holding world then the holding world documentation should be marked as such curtailing the Appearance Point development process and moving the primary development path to the Realisation Point.

The initial process of the Appearance Point requires a review of the World Specification Document for the world under consideration and the role of the world within inline hierarchy as presented in the Universe Specification Document.

In addition to this information supplemental data may be available from the Realisation Point for the world after the initial circuit.

Once the appearance requirements of the world under consideration have been clearly established the process of developing the graphical world object(s) can commence. Such development may be conducted in order to generate a new appearance node world or may make use of pre-generated objects from the World Resource Library as identified in the World Specification Document for the world under consideration.

It should also be noted that some degree of assimilation with the holding world's behavioural node may be required at this point due to the way in which VRML behaviour scripts directly manipulate the graphical objects. Where such direct manipulation is necessary the behaviour node is treated as the holding world for the appearance node in question effectively flattening the inline tree.

It is at this point that worlds considered for forward planning at the Requirements Point enter the appearance development process. Worlds entering the Appearance Point development process by this means should be considered as prototype worlds until ratified by the Evaluation Point processes once the primary development path is restored.

Once the appearance node has been developed the possibility for forward planning in order to prototype assimilation of the node within its immediate inline hierarchy may be considered. Such forward planning of assimilation can only be considered if the appearance node holding world's World Specification Document indicates that all subordinate node worlds and external files are available for assimilation. This option will generally only be taken where such holding world components have already been developed or are available from the World Resource Library.

If the forward planning option is taken primary development path is temporarily broken and further development of appearance node is moved to the Realisation Point. As with forward planning within the Requirement Point a world taking the forward planning route is considered as a prototype assimilation that will be refined when the primary development path enters the Realisation Point proper.

Having produced the appearance node and possibly explored assimilation the development must be recorded in the appropriate project documents. The documentation should include the nature of the graphical solution implemented, this should include the VRML geometry node types, and any considered but discarded solutions or techniques. This is required as node types may be available for further optimisation at the Realisation Point or should the implemented solution fail assimilation at the providing immediate ideas for corrective development routes.

Using the documented solution the derived appearance node should be examined in order to ascertain if it possible to further decomposed the graphical appearance with inline primitives or subordinate appearance nodes. If such decomposition is possible then the new inline nodes must be prototyped and the initial review point returned to establishing the appearance requirements for those new worlds.

On the completion of the appearance development loop a final review of the refined World Specification Document for the world under consideration and its role within the inline hierarchy as indicated by the Universe Specification Document must be held. The purpose of this review is to establish that all of the requirements detailed in the World Specification Document have been addressed to an appropriate level. If the review identifies that the appearance of the node has not been completed to a satisfactory level as indicated then the unsuccessful implementation may be attributed to a number of factors;

1. The allocation of resources to the node development may have been inappropriate or under subscribed.

2. The information contained within the World Specification Document for the node may be erroneous, misleading or vague.

3. The appearance as required by the World Specification Document may be unrealisable or unsupported and alternative solutions may need to be sought.

A node world failing this review point may be considered as an exploratory prototype world but the Evaluation Point must be returned to in order that that the World Specification Document and resource allocation may be re-evaluated. Any subsequent development for the world must be halted and the re-evaluated world is treated as a new development thereby allowing the development process starting at the *Behaviour Point* to be re-entered at the starting point.

The requirement that development for the re-evaluated appearance node is commenced at the Behaviour Point is specified as the behavioural characteristics of the holding world may directly affect the appearance node thus causing it to fail the review. Clearly if the developed appearance node is not dependant on its companion behaviour node world the Behavioural Point development process can be shortened and the existing node or the optimised node from the World Resource Library be used.

Those worlds passing the review point may now be considered as being complete and should be documented as such within the World Specification Document, the Universe Specification Document and the allocated resources within the Universal Project Plan be returned to the project resource pool for reallocation.

The completed appearance node world may now be considered for inclusion into the World Resource Library. In order to maintain the quality content of the World Resource Library a candidate object for submission should fit into one of the following categories.

1. The developed appearance technique represents a “standard” appearance strategy that can be widely reused.

2. The developed appearance is considered to be a new primitive shape. In such a case the use of the VRML PROTO node will be used to make the shape a true primitive.

3. The developed appearance is a highly complex node that would take much time and resources to recreate.

4. The developed appearance offers an alternative graphical technique to an existing node world within the World Resource Library.

If the option for inclusion is taken then a copy the appearance node world and its associated World Specification Document must be passed to the librarian for optimisation and clarification of documentation. A node world submitted to the World Resource Library may continue along the development process until the Realisation Point where the copy node may be substituted for the purpose of assimilation if its submission to the World Resource Library has been successful.

The conclusion of the Appearance Point development process requires that all processes initialised by the lines of concern from external points now be terminated and the primary development path be restored. The Appearance Point will have produced the required appearance node world in accordance to the World Specification Document that may be used in demonstrations to the client if required. The Appearance Point development must be signed of as complete before progression is possible to the Realisation Point.

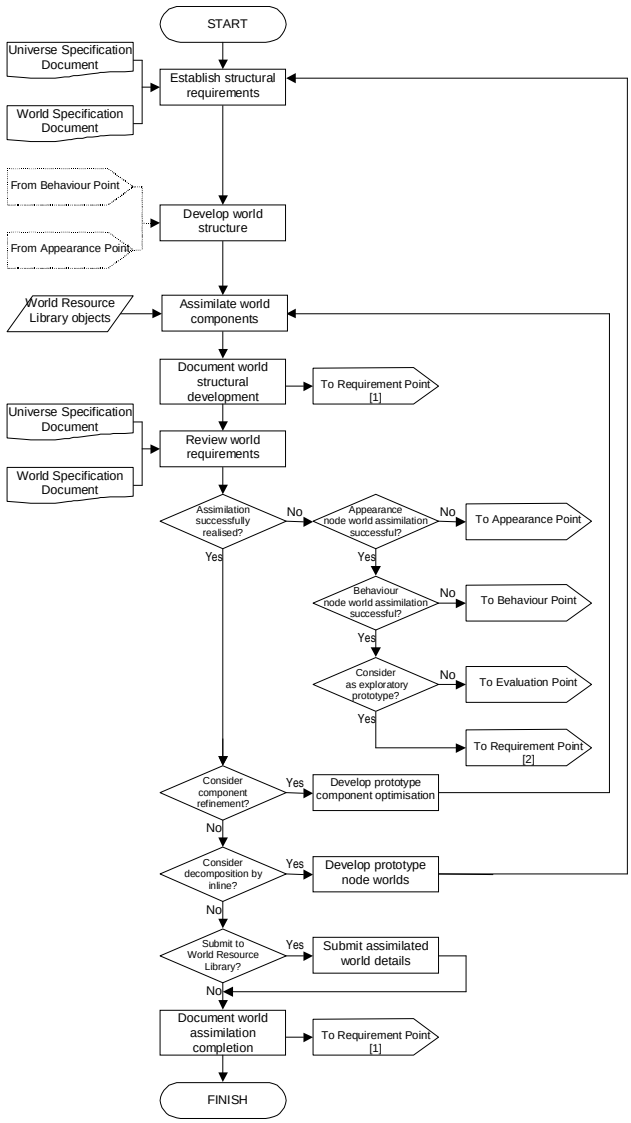
4.6 The Pentacle Method: The Realisation Point Process.

The Realisation Point of the Pentacle Method is concerned with unifying the different node worlds of a holding world according to the information held within the World Specification Document with the inline hierarchy as identified by the Universe Specification Document. The Realisation Point also provides processes to further establish inline decomposition, optimise the world performance and determine the quality of the world under consideration in order to determine its inclusion into the final universe product.

Access to the Realisation Point by the primary development path is only possible once the Appearance Point development process has been completed. All worlds developed during the primary development path processes must pass through the Realisation Point, this includes those worlds possessing only one node characteristic.

The initial process of the Realisation point requires that a review of both the World Specification and Universe Specification Documents be held this is conducted in order to establish the structure of the world under consideration.

Figure 26: The Realisation Point Process.



The derivation of the world structure can be seen as two separate considerations, the physical structure of file location and the internal file structure identifying the order for which subordinate inline worlds will be loaded into the holding world.

The physical location of files will depend largely on the restrictions imposed by the client at the Requirements Point and as such will have been recorded in the Universe Specification Document for the project. These restrictions may include a number of issues such as;

1. The requirement for the universe world to be spread over different World Wide Web servers.
2. The utilisation existing worlds or external files from the World Wide Web.
3. The containment of the world within a predefined or existing directory structure.
4. Alternative routing for world components if a component is unavailable.

Where no such restrictions have been placed on the physical structure of the file location for a world it is generally recommended that the structure as depicted by the inline hierarchy is replicated. This provides a structure that is instantly recognisable for maintenance purposes and provides a degree of organisation within the storage media host. There are, however, a number occasions where this general rule may not be applicable.

1. Components that are accessed by multiple worlds within the universe may be placed together under a common subdirectory at the root of the universe web. This is a strategy to reduce the amount of physical space that the universe world consumes on the host media and ensures that a common component changes are reflected though out the entire universe and not within a single holding world.
2. Worlds having no subordinates may be considered for inclusion in the holding world directory in order to reduce the complexity of the directory file structure.

3. Files of a size less than the host operating systems directory structure size allocation may be grouped with their holding world where the size of the available space on the host media is critical.

4. The components of a holding world may be grouped together where the access time to the world is critical.

Complimentary to the physical structural issues is the internal file order by which the subordinate inline worlds and external file objects are called into the holding world scene. The significance of the inline order will primarily depend on the target browser that the universe world is being developed for and the way in which it constructs the visible scene. Scene construction by browsers falls into two general categories

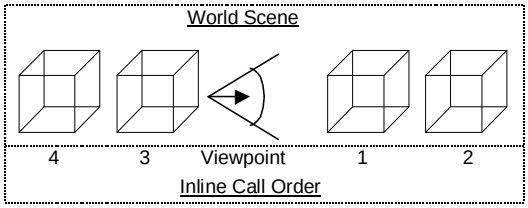
1. Incremental scene building, where the world scene construction is visible to the world participant. This is manifest as a series of visible wire frame boxes representing the inline files being “filled in” with the subordinate worlds as the VRML script is interpreted in real time.

2. Total scene construction, where the world scene is completely read from all inline components before rendering the scene to screen. This is manifest by a delay before the world participant is presented with the complete scene.

The ordering strategy will therefore depend on the target browser as indicated by the Universal Specification Document. Where a browser implements the incremental scene building strategy it may be possible for the world participant to commence interaction before the scene is totally completed potentially causing the unexpected “pop up” of worlds for complex scenes or slow rendering.

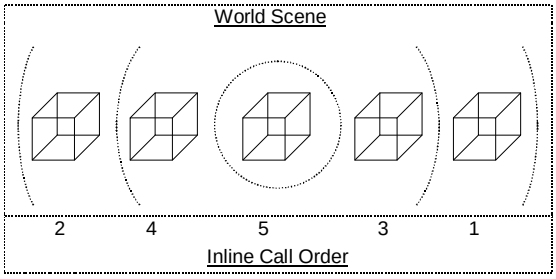
In order to overcome this the ordering of the inline world calls for the universe should be made in the reverse order of the world participants view, that is starting with the closest visible object and moving away the participants initial viewpoint (as shown in figure 27 opposite). For subordinate holding worlds this strategy may not always be applicable as the object may have been reused or instanced and require a different view of the object from the originally viewpoint developed for.

Figure 27: Universe World Inline Call Order.



In order to overcome this problem inline components for holding worlds should be called starting with the outermost objects and working towards the holding world origin. In keeping with the default VRML rule for construction of indexed sets objects at similar points from the holding world origin should be ordered in a clockwise direction along the Z axis. Clearly the start point of the ordering may not be reflected in the eventual ordering as a manipulated holding world is constructed. This strategy will cause the scene to be constructed with the central and potentially least visible object to be constructed last (as shown in figure 28 above).

Figure 28: Holding World Inline Call Order.



Having reviewed the available information for the world under consideration the world structure is developed. The process of developing the structure for the world under consideration will depend on both the physical location and call order for the world and the nature of the world, the universe world or a holding world. In addition to these considerations the development of the structure may have additional information provided by the development previously conducted as part of the forward planning of the Behaviour and Appearance Point process.

It is at this process that the forward planning route for node worlds from the Appearance and Behaviour Point enter the Realisation Point process for the development of prototype assimilation.

The development of the world structure provides the framework to allow the process of assimilating the disparate world components into the holding world. The assimilation process requires the component objects, or substitute objects if prototype assimilation is being undertaken, to be available either directly through the primary development path or from the World Resource Library.

The assimilation process requires that each of the inline components be called into the holding world and checked for compliance to their World Specification Document within the context of the holding world. Having tested each of the components individually a final assimilation for all of the holding world components must be conducted to ensure that they work together as specified by the holding world's World Specification Document.

This assimilation process must be conducted in the target environment as specified within the Universe Specification Document (platform, browser, host HTML application etc) and it is recommended that at least one other browser be used in order to identify problems with compatibility across platform implementations.

The process and strategies employed for both the structure and assimilation of the world under consideration must be appended to the World Specification Document by temporarily braking the primary development path and refining the project documentation by the Requirements Point processes.

Using the refined project documentation from the previous process the world under consideration may now be assessed for compliance to the World Specification Document. If the

world is found not to meet the required criteria during the review process then a number of pathways for corrective action are available in the following order.

1. For those worlds that have non-compliance to the World Specification Document against an appearance node world, the line of concern to the Appearance Point may be taken in order that the node can be reconsidered.
2. For those worlds that have non-compliance to the World Specification Document against a behaviour node world, the line of concern to the Behaviour Point may be taken in order that the node can be reconsidered.
3. For those worlds that have non-compliance to the World Specification Document against either or both node world types the world may be considered as an exploratory prototype. Where such a consideration is made the further development of the world is halted and the prototype world is used as a tool to elicit further information from the client at the Requirement Point.
4. If none of the above options are taken then the world and its associated documentation must be returned to the Evaluation Point for re-evaluation. Any subsequent development for the world must be halted and the re-evaluated world is treated as a new development thereby allowing the entire development process to be restarted.

If the world under consideration has successfully passed the review process being compliant to the World and Universe Specification Documents then the opportunity to refine the world may be presented. Optimisation of worlds may be conducted for a variety of reasons and utilise any number of VRML techniques to achieve this. Common techniques for optimising worlds that should be considered are.

1. The reduction of the detail level required by the world by use of the LOD separator to reduce initial rendering time.

2. The removal of redundant code, such as spurious comments, excessive white space and reduction of normals to a reasonable significant decimal point in order to reduce file size.

3. The combination of polygons to produce larger face areas to reduce rendering time and file size.

4. The uses of the DEF and USE separators as an alternative to inline for multiple instances of the same object in a world to reduce external file seek time.

5. The removal or pre-computation of light sources to speed up world participant interaction.

If the optimisation route it taken then a process of developing optimised prototypes is taken and fed back to the assimilation process. If the optimisation route is not required then the process of developing world structure and assimilation may have highlighted additional opportunities for further inline worlds to be considered.

Worlds considered as candidates for further decomposition by the inline strategy must have prototypes developed for each of the component inline worlds. Inline prototypes created by the Realisation Point process are not generally of either appearance or behaviour node world in nature rather they are refinements grouping objects into subordinate inline holding worlds. As such these prototypes are returned to the initial Realisation Point process in order to establish any additional structural requirements that may be necessary.

A world that has passed though all of the review points (either in its original form or recursively as an optimised prototype or new inline holding world) may now be considered for submission to the World Resource Library. In order to maintain the quality content of the World Resource Library a candidate world should fit into one of the following categories.

1. The developed world has been realised by a technique that represents a “standard” strategy that can be widely reused.

2. The developed world is considered to be a new primitive object (for example a chair, avatar, characteristic behaviour etc). In such a case the use of the VRML proto node will be used to make the object a true primitive.

3. The developed world is a highly complex node that would take much time and resources to recreate.

4. The developed world offers an alternative realisation to an existing world object within the World Resource Library.

If the option for inclusion is taken then a copy the world and its associated World Specification Document must be passed to the librarian for further optimisation and clarification of documentation. A holding world submitted to the World Resource Library may be carried along the development process until its own holding world reaches the Realisation Point where the copy node may be substituted for the purpose of assimilation if its submission to the World Resource Library has been successful.

The conclusion of the Realisation Point development process requires that all processes initialised by the lines of concern from external points now be terminated and the primary development path be restored. The Realisation Point will have produced the required holding world in accordance to the World Specification Document that may be used in demonstrations to the client if required. The Requirement Point development for the world under consideration must be signed of as complete before progression is possible to the Realisation Point as a component of its parent holding world.

Where the world under consideration is the universe world the development process is complete. All documentation surrounding the project should be reviewed to ensure there are no outstanding issues or developments that are still unresolved. It is recommended that in this instance the universe world be moved to the Evaluation Point for this final confirmation of completeness before publication.

4.7 Method Development Conclusion.

The development of the Pentacle Method from the original concept of synthesising the issues and concepts discussed in the previous chapters have far exceeded the original project objective. The movement away from simply developing a model to support the required to realise a VRML world has clearly been refined by the need to include both project management and quality assurance processes.

The actual derivation of the method has been built on the fundamental issues of methodology as discussed in chapter 3. These issues have been addressed by either directly by the consepts discussed in the latter part of that chapter or have been refined and clarified in order to present a more cohesive method of VRML world building.

Clearly as it stands, the Pentacle Method still represents a purely academic exercise in methodology construction and philosophy. In order for the method to be useful to world builders, and those involved within the world building project, the method must be evaluated. While the process of evaluation could be seen as an exercise in academic debate regarding the applicability of the conceptual nature of the method it is considered that empirical evidence is of far greater worth.

This consideration is based on the simple assumption that, if no existing paradigms support world building and the method can be shown to afford some support to the VRML development process, then this represents a significant improvement over the current state of affairs. This it is perceived will resolve the conjecture of the original hypothesis supported by the research and conclusions of chapter 2.

In order for this evaluation process to take place a test world will be generated against the Pentacle Method as presented, and the paradigms applicability to world building under VRML critically assessed in the following chapter.

5 Evaluation of Results.

"Even so faith, if it hath not works, is dead, being alone."

James 2:17. King James Version.

The previous chapters have provided two hypotheses postulating that current software engineering paradigms are not well suited to the requirements of VRML world building, and that until such a method exists the commercial world is unlikely to view VRML as a quality medium. To support the hypotheses current trends in use of current paradigms have been examined both from real world cases and academic viewpoint of applicability to the media. From this research it has been concluded that the hypotheses are supported. In order that the apparent lack of structured support for the process of VRML world building be redressed, the fundamental principles of methodologies have been explored and their application to a new paradigm been considered. These considerations have been refined and presented as the Pentacle Method, a new paradigm for the creation and management of VRML world building within the context of a managed project.

In order to establish the validity and applicability of the Pentacle Method as a tool to facilitate the building of quality VRML worlds it is proposed that an example world be built in accordance to the principles advocated by the method. It is considered that this is the most practical option to assess the method and that the evaluation process will compare and contrast the theoretical application of the method against the actual implementation of the example world. By such a process of comparing and contrasting it is expected that any flaws or potential refinements to the Pentacle Method as presented will be highlighted.

5.1 The Problem Domain.

In order to implement the Pentacle Method for the purpose of evaluation it is considered expedient that the examined problem domain be briefly discussed. The selection of an example problem domain that was suitable for the purpose of evaluation required that the scenario allowed for the following to be accommodated.

1. To demonstrate the graphical abilities of the VRML medium.
2. To demonstrate the application of behaviour to a world.
3. To demonstrate the software engineering aspect of VRML under the Pentacle Method.
4. To demonstrate the artistic aspect of VRML under the Pentacle Method.

In order for some kind of quality measure to be levelled at the scenario world to be generated it was also considered that the world should represent a realistic application of VRML as a medium for the dissemination of information. The selection of scenario therefore required a world that contained a number of factors that could be used to establish such quality, as follows.

1. To establish if the world imparted additional and significant value to the problem domain by enhancing the quality of information.
2. To establish if the world is “fit for use” as a measure of overall quality.

Previous research into the practical application of artificial reality as part of the collaborative paper referred to in chapter 1 indicated that the prime use of virtual reality per se is in the field of simulation, or rather making the normally unrealisable manifest. This combined with the author’s interest in archaeology and recent interest in the use of information technology within the world of restoration and conservation within museums, identified the field of archaeological artefact reconstruction as a prime candidate for the scenario world.

Originally it was hoped that an object could be secured to base the scenario on, however neither the museum approached or the school of applied art (conservation and restoration) approached for a specimen were unavailable or unwilling to release such an artefact for the purpose of the evaluation. It should be noted however that both expressed some interest in the project domain. As a consequence a replica of a clay pot in the Roman style was purchased from which to take geometry measurements. The reconstructed artefact therefore bares little no historical accuracy.

The (fictional) client requires that the following points be implemented within the final reconstruction.

1. The reconstruction must clearly show the form of the pot.
2. The reconstruction must show the detail of the available fragments (rim, side and base) in situ.
3. The reconstruction must allow for each of the fragments to be manipulated individually.
4. The reconstruction must provide mechanisms to access information HTML pages.
5. The reconstruction must provide mechanisms to access each individual fragment.
6. The reconstruction must be able to be viewed from a number of viewpoints.
7. Each of the fragments must be able to be viewed and manipulated in isolation.

The reconstruction is to be run across a local area network with the world files held on the museum server and displayed by an entry level PC with mouse at the exhibit location. The final reconstruction is to form a basis for an internal report on the possibilities of utilising the VRML medium as an alternative information access media for general public use (this is not required from the developers).

With these considerations in mind the scenario for evaluation requires that a Roman pot be reconstructed from the three existing fragments using VRML under the Pentacle Method

5.2 Initial Limitations and Assumptions of the Evaluation.

The previous section has provided a brief outline and justification for the scenario problem domain for which the Pentacle Method is to be evaluated by. It is noted however that there are a number of concerns regarding the limitations of the process of evaluating the method as presented in the previous chapter. In addition to these limitations the scenario as outlined above relies on a number of assumptions to be in place before the evaluation process can be undertaken. This section will, therefore, briefly discuss these perceived limitations and assumptions prior to the evaluation process.

The Pentacle Method has been designed for use within a project oriented environment requiring a number of active players at each point as shown in figure 29 opposite. While it is considered that the method may be employed by a gifted individual who has an appreciation of these disparate disciplines it is considered that an implementation of full project management techniques and quality assurance practices are beyond the scope of the projects brief.

Figure 29: Parties Concerned During Development.

	Project Manager	Client Representation	Quality Assessor	Requirements Analyst	Behavioural Engineer	Aesthetic Artist	World Builder	Resource Librarian
Requirement Point	Yes	Yes	No	Yes	Option	Option	Option	Yes
Evaluation Point	Yes	Option	Yes	No	No	No	No	Yes
Behaviour Point	Yes	Option	Option	No	Yes	No	No	Option
Appearance Point	Yes	Option	Option	No	No	Yes	No	Option
Realisation Point	Yes	Option	Option	Option	Option	Option	Yes	Yes

The absence of a real client and external management further restricts the method as prototypes and point deliverables cannot be used to solicit more information at the Requirements Point or establish the nature of the decision-making processes. Because the author is in effect playing a multitude of differing roles with differing objectives and spheres of influence the results of the evaluation may be open to a degree of subjectivity in their analysis. Therefore, the quality of the

evaluation must be considered as if not dubious coloured by the authors proximity to the project domain.

The Pentacle Method requires that each world that is discovered undergo a cycle of the primary development path in order for the world building process to be achieved in a managed manner. While ideally the evaluation process would document the development all of the worlds discovered the space allocated within this paper prohibits such an extensive record. It is therefore considered that only an overview of the development via the primary development path and the lines of concern can be shown in the documentation of the evaluation process. This, of course, does not imply the scenario world will be developed as an ad hoc process, for such development would totally invalidate the evaluation of the method. It is hoped that this strategy will encapsulate the salient development processes of the method and provide significant information to show the reader that evaluation has been conducted.

The evaluation process is clearly intended to assess the applicability and quality of the Pentacle Method as a tool to enhance the process of VRML world building. This said the quality of the evaluation material, that is the world to be produced from the scenario problem domain, will directly reflect the abilities of the author with the VRML and Java Script media. While the author considers that he has a degree of competence with the VRML language it is freely conceded that he is no expert within the VRML community. The derived solution as input for the evaluation process of the method may therefore be an imperfect implementation of the language and subsequent evaluation of the method from this material must be considered in this light.

While these restrictions arguably devalue the overall assessment of the method, the remainder represent the core features of the method, namely the recursive development and refinement of worlds via the primary development path, and the iteration afforded between the five points by the connecting lines of concern. It is therefore considered that if the concept of such interplay

between points, by either access method, can be demonstrated to be of significant worth then the marginalised issues can be seen as adding value to the method as a whole unevaluated.

The scenario as presented in the previous section requires that some original, and perhaps unrealistic, assumptions be made before the evaluation process can commence. These assumptions are as follows.

1. The client has a clear understanding of what the VRML medium can achieve.
2. The client has a clear understanding of what they require.
3. The client requires no specialised equipment or devices to be employed.
4. The client has the IT infrastructure in place to realise the project.
5. The client has no preferred browser or development tool requirements.
6. The client is able to provide geometry for the world objects.
7. The client is able to provide images for texture mapping.
8. The client is able to provide HTML pages for linking worlds to.
9. The client has no existing structure or structural requirements for the world.

With these initial limitations and assumptions stated the process of developing the solution to the scenario problem domain in order to evaluate the applicability of the Pentacle Method is described in the following section.

5.3 Overview of Development Using the Pentacle Method.

The overview of the development process is intended to evaluate the applicability of the Pentacle Method derived in chapter 3 and presented in chapter 4 as a supporting software engineering paradigm for VRML world building. The development of the scenario world as described in the previous section was conducted using the method within the confines of the limitations and assumptions as stated.

The initial process of the requirement point established the world requirement as stated in the previous section as providing.

1. A graphical representation of a Roman cooking pot based on available geometry.
2. A graphical representation of three fragments from available geometry.
3. A device or devices to permit access to information in available HTML format.
4. A device or devices to permit interaction with the identified components.
5. A range of different viewpoint that permits the main pot to be viewed from different vantage points.

A search of the World Resource Library (represented by research and experiment amassed during research for the previous chapters) identified a number of existing worlds that had potential for reuse.

1. A behaviour node world allowing the world participant to click and drag specified objects along the world X and Y axis.
2. A behaviour node world allowing the world participant to click on a specified object causing it to rotate in a predefined manner.
3. An extrusion node implementation that allows a specified 2D co-ordinate plain to be extruded and rotated around a specified point.

4. A billboard node, which keeps the specified group objects, facing the world participant view.

The resource allocation for the project as required by the method indicated that a number of tools were available to the project from previous experimentation with the medium. These were identified as.

1. 3D Design Plus : A geometry modeller.
2. SitePad : A VRML 2 file editor.
3. VRML Express : A VRML 1 file editor.
4. VRML1TO2.EXE : A command line VRML specification conversion program.
5. WCVT2POV : A mulit 3D graphic file format converter.

Human resource allocation to the project was necessarily one member of staff (the author) who would be responsible for all aspects of the project development. The hardware requirements, unspecified by the client, for development purposes were identified as two entry level Pentium PC's (one acting as the server) running across a TCP/IP Intranet under MS Windows 95. The choice of hardware, available software and VRML 2 requirements restricted the availability of the target browser to be developed against. The final selected application was SGI's CosmoPlayer a plug-in for MS Internet Explorer, with Sony's Community Place stand alone viewer for testing and conformance comparison. It was considered that this suite of resources was sufficient for the development of the world.

The process of documenting the project specification and the single world requirements thus far discovered was conducted being recorded in the Universe Specification Document and World Specification Document for the universe world ROMAN_POT, respectively. As previously noted the project management device requirements were not implemented but the existence of the Universal Project Plan is acknowledged for completeness.

The opportunity afforded by the connecting lines of concern to develop exploratory prototypes for both the required behaviour and appearance nodes of ROMAN_POT was taken.

The prototype development of the behavioural node commenced with a process of establishing the node requirements, in accordance to the method. It was apparent that the required behaviour for the universe world fell into two distinct areas, lighting and viewpoint.

A number of prototypes experimenting with different lighting effects and viewpoint positioning were generated and tested against a primitive sphere world pulled from the World Resource Library. It was concluded from these experiments that a single directional light source was adequate for the world lighting requirements and that six viewpoints (top, bottom, front, reverse, left and right) would be sufficient to cover all the cardinal points. These conclusions were documented generating a new World Specification Document for GLOBAL_BEHAVIOUR node world.

The presented option for inline decomposition clearly was advantageous, as there were two distinct behavioural requirements, lighting and viewpoint. Prototypes for each of these worlds were generated from the information within the World Specification Document for GLOBAL_BEHAVIOUR node world named LIGHTING and VIEWPOINT respectively. These worlds were documented within their own World Specification Documents and as no further inline decomposition was considered at this stage added to the inline hierarchy diagram in the Universal Specification Document.

The prototype development of the appearance node commenced with the process of establishing the node requirements in accordance to the methods. It was apparent that there were four distinct areas of concern, the three fragment objects and the pot reconstruction. It was noted that behavioural requirements also existed within the domain of the appearance node specifically the hyperlink and interaction requirements for the fragments.

A number of prototypes for the node were generated using the geometry provided by the client and a combination of the geometry modeller and file converter. From these experiments it was concluded that the appearance node world could not be realised purely as a single object if the requirements were to be accommodated successfully. These results were recorded in a new World Specification Document RECONSTRUCTION node world.

As with the behaviour node of ROMAN_POT the opportunity for inline decomposition was clear as four distinct areas of concern had presented themselves in the three fragments and the pot reconstruction itself. Prototypes were generated for the fragments and the pot from the requirements recorded in RECONSTRUCTION World Specification Document. These worlds were documented within their own World Specification Documents as POT_RIM, POT_SIDE and POT_BASE for the respective fragments and POT_RECON for the reconstruction. Further inline decomposition was not considered necessary at this stage and the newly discovered nodes added to the inline hierarchy diagram in the Universal Specification Document.

The conclusion of the prototyping diversion via respective the lines of concern returned control to the primary development path. As no client was present in the evaluation process by the nature of the project the primary development path was progressed to the Evaluation Point.

It is noted that at this point within the development under the Pentacle Model the worlds discovered during the exploration of the ROMAN_POT.WRL will commence their own primary development paths. It is not considered feasible to show this process for the reasons identified in the previous section. In order to show the method in action, however the development of ROMAN_POT will be continued under the assumption that these worlds are being developed in parallel.

In accordance to the method a review of the existing project documentation was conducted in order to establish ROMAN_POT world requirements and review the information discovered thus far. From this review it was considered that ROMAN_POT required no inherent behaviour

or appearance, these being conducted in the inline node worlds GLOBAL_BEHAVIOUR and RECONSTRUCTION. This identified ROMAN_POT not only as the universe world but also as a holding node world.

By virtue of the forward planning of the world conducted at the Requirement Point it was considered that no further inline decomposition for ROMAN_POT was required and that the information contained within the project documentation was sufficient to permit the continuing development of the world. These observations were recorded on the World Specification document for the world.

A cursory review of the objects within the World Resource Library identified (unsurprisingly) that no previous worlds had been generated that matched the requirements of ROMAN_POT. It was also considered that, as the universe world, no dynamic internal behavioural attributes were required (these being conducted by the inline behaviour node GLOBAL_BEHAVIOUR).

These observations concluded that the VRML specification for the development of the world should be version 1 and could be realised given the resource allocations identified at the Requirement Point. These observations and considerations were noted on the World Specification Document and the primary development path moved to the Behaviour Point.

In accordance to the method the Behaviour Point process commenced with the review of the world requirements for ROMAN_POT world. As identified in World Specification Document ROMAN_POT represented the universe world and a holding world. This noted the requirements of the world were therefore to provide hyperlinks to the two inline worlds GLOBAL_BEHAVIOUR and RECONSTRUCTION.

The development of the world therefore constituted the creation of two, grouped inline nodes to these as yet unavailable files. In order to establish these files within an accessible structure the

opportunity for forward planning of prototype assimilation was taken by the line of concern extended to the Realisation Point.

The forward planning route to the Realisation Point commenced with the establishment of a structure for the world. As no preference of restrictions had been placed on the structure of the world development (identified in the Universe Specification Document form the noted assumptions) a replication of the inline hierarchy as recommended by the method was considered to be appropriate. As GLOBAL_BEHAVIOUR would be a common node to all of the components of ROMAN_POT this structure was refined, in compliance to the method recommendations, to provide two physical subdirectories, reconstruction and common, from the project root directory.

As none of these worlds had been finalised the assimilation process utilised the developed prototypes for the lighting and viewpoint worlds and a sphere representing RECONSTRUCTION from the World Resource Library. These exploratory prototype worlds were appropriately named and placed in the respective directories. In compliance with the method this was noted in order that the prototype worlds used would not be accidentally assimilated into the finalised universe.

A review of the project documentation and the prototype development identified a minor problem with the assimilation of the world components. This concerned the initial placement of the viewpoint on the world scene. The prototype assimilation under CosmoPlayer placed the initial viewpoint at the centre of the world co-ordinates (the world origin) at the same point at which the substitute RECONSTRUCTION was placed by the inline call. Comparison with Community Place browser noted that this did not occur, the original viewpoint defaulting to the first viewpoint node encountered.

As CosmoPlayer was the target browser for the project it was perceived that this issue was of some significance and clearly remedial action would need to be taken. The option to refine the

prototype was taken returning to the initial process in order to establish what these requirements might be. Further experimental prototype development concluded that the inclusion of an arbitrary viewpoint in ROMAN_POT world providing a default world participant view was required. This conclusion and experimentation was recorded on the World Specification Document and no further options were taken restoring the primary development path.

It was considered that the further decomposition of the world was not possible as both appearance and behaviour nodes for ROMAN_POT had already been established. Clearly the newly discovered requirement for a default viewpoint could not be considered as an inline as prototype assimilation had proved that this was not an option. The world was reviewed against the requirements and it was considered that the developed behaviour matched the specification. A note was made on the World Specification Document that the arbitrary viewpoint assigned to the world during the prototype assimilation would need correction when the viewpoints were established during the development of the VEIWPOINT world.

Because of the nature of the world, being a unique universe world, and the fact that optimisation was required the world was not considered for submission to the World Resource Library. This concluded the Behaviour Point development process for ROMAN_POT and control was passed to the Appearance Point following the primary development path.

In accordance with the method an initial process of reviewing the project documentation was conducted. It was noted that ROMAN_POT was a holding world with no graphical requirements, in accordance to the provisions of the method for such cases the development process was skipped and signed off as not being required on the World Specification Document. The primary development path was therefore progressed to the Realisation Point.

The initial processes of the Realisation Point of establishment and development of a world structure was by passed as this had already been established within the World Specification Document by virtue of the forward planning option taken at the Behaviour Point. As no

finalised inline components were ready for assimilation ROMAN_POT world development was suspended until these inline nodes were completed.

It is noted that the development of these absent worlds have been conducted in parallel with the development of ROMAN_POT each world entering its own development cycle along the primary development path and development conducted in a similar manner to the example of ROMAN_POT provided. In order for the example to be completed it is therefore assumed that this period of parallel development has taken place and that the inline worlds are now available

According to the methodology, the assimilation of GLOBAL_BEHAVIOUR and RECONSTRUCTION into ROMAN_POT was conducted. This was documented and the following review held in order to establish if the world met the requirements as detailed in the project documentation. This review concluded that the world complied with the project documentation and the option to refine the world was considered.

As previously noted the default viewpoint for the world was still unrefined. Using the forward viewpoint developed as part of VIEWPOINT world the default world participant viewpoint was amended. It was further noted that the GLOBAL_BEHAVIOUR world was a pure holding world and as such it was considered that it could be removed within ROMAN_POT as an inline and the two components LIGHT_DIRECT1 (the refined lighting behaviour node world) and VIEWPOINT substituted. In accordance to the method a prototype was developed to reflect these two refinements and the Realisation Point of assimilation looped back to.

The subsequent review established that the new ROMAN_POT world functioned as required and that no further refinement was possible. It was also considered that no further decomposition was available and the respective world documentation amended to that effect.

As this is the universal world a final sweep of the primary development path is recommended by the method in order to finalise any outstanding issues or address any concerns that have not yet been rectified. This process was conducted and the world signed off as complete.

While the example development as presented above represents a brief overview of the construction process for the universe world, it is considered that it does encapsulate the salient features and spirit of the Pentacle Method as presented. Clearly the example development requires that the missing parts of the development process, most notably the Appearance Point processes and the management and quality assurance aspects of the method (as previously noted), be taken on faith. This only serves to further muddy the evidence provided by the evaluation process.

5.4 Conclusions of Evaluation.

The theoretical production of VRML worlds using the Pentacle Method was conducted during the development of the method by a process of role playing indicated that the method could prove useful to a project oriented VRML development. The reality of implementing the method as a single developer and with the limitations and assumptions previously stated provided a less than perfect result on which to base the evaluation process on.

The evaluation of the Pentacle Method, as presented in chapter 4, with regard to its application to solve the scenario problem domain, highlighted a number of concerns and issues. These are, in no particular order of significance, as follows.

5.4.1 The Primary Development Path.

The author still has some reservations with regard to the order of the Behaviour and Appearance Points within the methods primary development path. While the process appears to work within the general primary development path as identified in chapter 3 there is some question as to using the same order for the advance planning option within the Requirement Point. Punitive tests of reordering these two development point on the primary development path identify that while the generation of visible objects provides a more rapid result for evaluation associated behaviour becomes more difficult to realise because of additional the complexity associated with the appearance node.

It is the opinion of the author that as the generation of behaviour under Java Script is more involved than the production of pure VRML objects that the proposed order is correct. This is based on the observation that less time and effort is required to fix a world fault that is founded on behaviour than appearance as, by the nature of the media, the fault will generally be manifested visually. It is conceded that this is the opinion of the author who has more

experience with pure VRML than Java Script and it is therefore suggested that the roles may be interchangeable depending on the skill set weighting available to the project.

The method makes reference to the top down derivation of world requirements and the construction of the world by a bottom up process. The method text and flow charts indicate this process by the decomposition of the world under consideration by the opportunity to inline components (Top down design) and the assimilation process at the Realisation point which requires that all the world subordinate components are present (Bottom up construction).

While the top down and bottom up development process is implied in the narrative of the method text, there is no direct implementation of these strategies. The use of the inline hierarchy diagram notation clearly makes provision for addressing this problem, to some extent, by identifying completed worlds, represented as filled icons, and prohibiting higher level worlds completion until all the inline nodes are themselves complete. The order in which these worlds are processed, however, is purely arbitrary and relies on the documentation process for the documented world to reflect any changes in higher order worlds. It is considered that these strategies require further enforcement within the method if the learnability concerns discussed in chapter 3 are to be readily addressed.

5.4.2 The Lines of Concern.

It is implied within the method that lines of concern may be followed during the development a prototype node under the forward planning provisions. Such a process may well prove useful to the developer concerned, especially if they are multi-skilled personnel, but has an inherent danger of getting lost down a metaphorical “rabbit hole” as prototype development passes arbitrarily between point processes along connecting lines of concern. While the method makes such paths available to the developer it is not recommended that the development of prototypes be conducted in such an unmanaged way as this will ultimately lead to prototypes being built on prototypes. Such a situation would clearly degrade the quality of development with each successive stage within the prototype development process.

It is suggested that if such a route is taken between points then strict time allowances must be imposed by the project manager for any prototype discovery process conducted in this manner. In order to prevent the rabbit hole syndrome any prototype development using this strategy *must* be terminated upon its expiry regardless of the point reached and the primary development path restored.

It is further noted that the method calls for all process instigated by lines of concern be terminated at the conclusion of the development point. This, it is considered, is only relevant to the Evaluation Point where processes may need to run to the point terminator. By their nature prototypes developed for the Behaviour, Appearance and Realisation points are exploratory and therefore only representative of ideas to be considered when the primary development path accesses the respective point, this is why the prototype development process is called forward *planning* and not forward *development*.

There is little justification, therefore, in progressing a prototype beyond the documentation of the development thus far conducted for the reasons regarding quality as discussed above. As all documentation is conducted by a following the line of concern from any point to the Requirement Point in order to check that developments do not impact on other work, it would seem logical that the termination of prototype development should be at this point.

The method requires that all processes instigated by lines of concern be returned to the primary development path on termination to the calling process. It is noted that for some processes, particularly those calling the Evaluation Point, this represents an inefficiency in the development process as the additional information gathered during the external point process is not assimilated into the node as it is outside the development loop for that point. While the method makes provision for this by considering the world to be a prototype, it must complete another cycle of the primary development path for the new information to be acted upon.

While this is not unreasonable, based on the view that if one piece of information is missing then the chances are that other information is too, the method treats minor both minor and major alterations alike. In order to overcome this perceived inefficiently it is suggested that there should be provision for the integration of any additional requirements toward the end of the calling point process. It would therefore appear that the most logical place for this option would be immediately after the jump to the Evaluation Point is made instigating a loop back to the initial process of the respective point.

5.4.3 The Inline Hierarchy Notation.

While the inline hierarchy notation provides a clear and highly visible representation of the project development there are two concerns noted by the author. The first of these is the use of inline hierarchy diagrams within the World Specification Document. The method narrative implies that the whole of the inline hierarchy is required in order to establish the position of the world under consideration. While such an expanded view may prove useful the practicality of both time limitations and space available on the physical document makes this practice impractical. It is suggested that the inline hierarchy diagram contained within a World Specification Document detail only those connected objects *immediately* above and below the world under consideration. If further clarification of the world position is required then the full inline hierarchy diagram of Universe Specification Document should be referred to.

The other concern noted is the use of the notation to include identified and explored worlds. As these two types of worlds are discovered it is clearly advantageous to include these into the inline hierarchy diagrams but there may be no indication as to the nature of the node world (holding, behaviour or appearance) or to the VRML specification to be used. It is suggested that all worlds, for which these issues have not been resolved, be provisionally identified as *holding node worlds* using the VRML 2 specification for development. This, it is considered, represents the “worst case” scenario that should be planned for until the world is revisited for refinement by the primary development process.

5.4.4 Pure Holding Node Worlds.

The method makes no direct reference to the point at which a pure holding world, a world with no internal behavioural or appearance attributes, is constructed. While the narrative identifies the VRML inline node as a static behaviour and therefore such a world should be constructed at the Behaviour Point, this, it is felt, is not strongly enough emphasised within the method.

It is noted that the development of pure holding node worlds represents a trade off between the efficiency and effectiveness of the universe world. The inclusion of pure holding worlds clearly aids the structure of the world both in terms of world development and the appearance of the time to screen rendering for incremental scene building browsers and in providing prophylactic manipulation over grouped objects. However, it is observed this has a slight impact on the down load time as the holding world is interpreted to find its subordinate inline components. Of more concern is the case where a holding world is unavailable for some reason, this will result in all of the inline nodes not being called into the world scene even if the access to the individual components directly is possible.

It must therefore be considered that further optimisation of the world could be achieved by the removal of redundant pure holding node worlds. In order to derive a flexible approach to the use of pure holding worlds this decision would clearly depend on a number of factors such as client requirements (speed, fault tolerance etc), target browser, required behaviours, confidence in world availability, ability to document world removal clearly for maintenance purposes etc,

5.4.5 Java Script and Pure VRML Behaviour Node Worlds.

The method makes no clear distinction between behaviour node worlds that have been created using Java Script and those that utilise native VRML functionality. This lack of distinction may potentially lead to confusion over documentation (although it is noted that the resulting world would still exhibit the behaviour) and misallocation of human resources (graphic artists conducting behavioural engineering work and visa versa). An example of where such confusion

may occur is in the implementation of the VRML 2 extrusion node which requires no actual script to be created because this functionality been abstracted within the implementation of the language to a more developer friendly format.

The definition of what constitutes a behaviour node world is far beyond the scope of this project's brief. In order, however, to address this potential confusion to some degree it is the opinion of the author that a native VRML behaviour node that produces a visual object and that cannot be decomposed by a process of inline should be considered as an appearance node. Conversely a Java Script implementation that manipulates primitives such as point sets in order to form a new object that is solely graphical in its nature should also be considered as an appearance node.

In this regard, pure VRML nodes that have no physical manifestation but rather compliment another object must therefore be considered behaviour nodes.

5.4.6 The Requirement Point.

The "establish project resource allocation" process identifies project resources as an input to the primary development path. Although the need for this is clear, the location of the information is somewhat nebulous being referred to as simply "project resources". If these resources are to be successfully managed then some form of co-ordinating body must be required to facilitate this role. It is suggested that this management be placed within the domain of the World Resource Library.

This will expand the scope of the librarians brief beyond the husbandry of worlds to include management of tools, hardware, other software and personnel details. Clearly such an expansion of the role of the librarian impacts on the availability of that member of staff for development suggesting that the role requires a dedicated member of staff disciplined in all of the aspects of world building or for a larger project possibly a small department.

The production of the Universal Project Plan assumes that the project manager will implement a range of “standard” devices in order to assert control over the development project. Research into the area of project management indicates that only 20% of project managers are “veterans” of their art (Allison (1996)). If this is the case then it may be prudent for the method to include a review of project management devices and strategies before their commitment to the Universal Project Plan.

While the inclusion of such a process within the process of the method is mute, it would provide a tailor made project management plan for each universal world built as opposed to using a “standard” format of management strategies. This would ensure that the devices employed by the manager could be specifically targeted at the problem domain and be applicable to the various tasks required by the method. It is therefore considered that such an additional review would instil additional quality into the world building process redressing the concern of the second hypothesis of this paper.

The conclusion of the documentation process moves straight into the decision point for forward planning of prototypes. While it is indicated within the method text that this option is included to capitalise on the impetus generated from the exploration of the world requirements and pre-empts client acceptance of documentation, no reference to the submission of documentation to the client is shown on the point process flowchart. This is clearly an oversight in the learnability issue of the method; potentially a user of the method could enter a prototyping phase before the submission of the documentation to the client if the flowchart were to be solely used. Clearly if the client were to reject the proposals of the documentation time would have been wasted on constructing inappropriate prototypes.

It is considered that this could be rectified by the addition of a new process “Submit Documentation for approval” with a companion decision point “Documentation accepted?” on the flowchart. The acceptance decision point paths would lead to the terminator if the documentation was accepted or to the forward planning option if the client had not yet accepted

the proposed development. In this case the negative path represents the *delay* between submission and acceptance, *not* the fact that the documentation has been rejected, which case is addressed by the precondition of the Evaluation Point.

5.4.7 The Evaluation Point.

The order of the two initial decision points, the option to decompose by inline and the amount of information available to continue development is questionable. If the world can be decomposed and the requirement review undertaken there is the potential for an absence of vital data, this could subsequently result in further decomposition against an erroneous requirement review. This loop becomes far more effective as a quality assurance device when the order of the points are reversed providing the assessment of the value of information available before the consideration of inline. However, such a reordering of the points may be less *efficient* with regard to the speed of development as the Requirement Point must be returned to if the world information is considered insufficient.

The design therefore is to either remain with the order as presented, representing a number of smaller problems with an imperfect requirement specification, or with the proposed order representing a large problem domain for which the information is complete. It is suggested that the former, original, ordering represent the best option. This view is a mute one but in its defence it should be noted that even if the information for an inline node world were incomplete or erroneous it would be reviewed at either the Behaviour or Appearance Point, depending on the node type. If the latter order was in place this would purely be a duplication of effort as the review process for the development point will reassess the requirements at a more *timely* point within the primary development path.

The Evaluation Point flowchart appears to indicate that the method requires that if no dynamic is present then the VRML 1 specification should be used. This is clearly not so, rather represents an opportunity for the project manager to consider allocating human resources to the

development under the VRML 1 specification. This decision point is included in the method in order provide flexibility to the issues surrounding development staff who may be well versed in the original specification but unfamiliar with the newer specification which supports dynamic behaviour using Java Script.

The VRML 2 specification also includes additional nodes or node refinements that are unavailable in the original specification that are not of a global or behavioural type, such as the EXTERNPROTO. Development of such nodes under VRML 1 at best represents a waste of resources and inefficient execution as they exist as primitive nodes of the newer specification and at worse represent pure folly, as they simply cannot be replicated. The key purpose of this point (and indeed the whole method) is the provision of a framework for the development process to be conducted within not to and therefore the project managers common sense is required to establish the VRML platform to be used.

The author also considers that while this decision point to select the VRML development platform is correct at the time of writing it will become out of date as future refinements of the specification inevitably are released. As these future refinements are unknown the platform selection cannot be based purely on the behavioural requirement of the world. It is therefore suggested that the decision point should be based around the features implemented in the latest specification that are unavailable in older releases (this of course is primarily the inclusion of behaviour in VRML 2).

5.4.8 The Behaviour Point.

The method does not specifically identify the viewpoint, lighting, sound and other VRML node types as being under the behaviour node world group (as noted above). While such nodes could be argued as belonging to the appearance node it is considered that these *effect* the appearance of an object rather than constitute the appearance of the object within the appearance node world. This view is further supported by the nature of such nodes that are generally applied

within the method as *global* effectors in order to manage such node implementation in a realistic way.

The positioning of the node development process before the opportunity to consider decomposition suggests that a node *must* be constructed before decomposition is possible. If the node has clear opportunity for decomposition from the outset then there would seem to be little justification in developing a world that will ultimately become a pure holding world for component inline nodes.

The processes were originally placed in this order to aid the behavioural to discover opportunities for inline nodes through the process of developing the world under consideration. If the order is reversed this opportunity is lost, as is the opportunity for forward planning of assimilation, clearly a disadvantage. It is suggested that a solution to this may be found in the reordering of the processes, in order to pre-empt construction by considering inline development, and a new loop from the subsequent development process back to the initial establishment of requirements process. This in effect shortens the development loop by removing the “create and explore prototypes” process leading from the inline consideration and replacing this with the actual development process.

This however makes the not unreasonable assumption that behaviour nodes should be considered as prototypes until no further inline decomposition producing the most cohesive and loosely coupled nodes is possible. The reordering of these processes from either development point will restrict the ability for the node world to be considered for forward planning of assimilation. In fact far from being a restriction it is perceived that this would *tighten* the quality of the world building process. This view is based on the observation that prototypes sent for test assimilation would be built on *pre-acceptance* nodes (that is prototypes that have been fully realised but not signed off as completed) rather than purely experimental prototypes

5.4.9 The Appearance Point.

Similar to the concern noted above in the Behaviour Point evaluation, the positioning of the node development process before the opportunity to consider decomposition suggests that a node *must* be constructed before decomposition is possible. As noted, if the node has clear opportunity for decomposition (highly likely if primitives are being used) from the outset then there would seem to be little point in developing a world that will ultimately become a pure holding world for component inline nodes.

The processes were originally placed in this order to aid the graphic artist to discover opportunities for inline nodes through the process of constructing the world appearance. As with the Behaviour Point reversing the order of these processes denies the developer the opportunity for discovery and prohibits the early opportunity for forward planning of assimilation. A similar solution to the Behaviour Point as noted above, requiring the reordering of the processes may resolve this problem.

This however enforces the assumption that *all* developed nodes (not just behaviour node worlds) will be considered as prototypes until no further inline decomposition is required or appropriate. While the benefits of reversing the order for the Behaviour Point are valid there must be some question as to the appropriateness of applying such a regime on the developers artistic ability. It should be borne in mind that the appearance node will, at some point, pass throughout the Realisation point optimisation processes where the opportunity to decompose a complex world is offered once again. It is therefore suggested that there may be significant benefits in letting the graphic artist construct the world en toto before they consider decomposition.

5.4.10 The Realisation Point.

The initial process of establishing the world structure must clearly be conducted before any form of assimilation can take place. However the method requires that the entire assimilation

process be conducted before the option to decompose by inline is considered. Clearly, as with the Appearance and Behaviour Point processes there seems little point in conducting this entire assimilation of worlds within a structure that ultimately will be discarded in favour of a updated one reflecting a new inline hierarchy.

It is considered that the developer must be afforded the opportunity to develop a structure in order to discover inline opportunities and therefore the initial two processes of establishing and developing the world structure must remain in their original places. It is suggested that the problem of late inline consideration could be overcome by moving the inline decomposition option to a point immediately after the production of the initial structure, creating a short loop back to the initial process. This would allow the discovery of inline worlds to take place during the structural development and be acted upon before assimilation is commenced. Further opportunities for inline decomposition that arise as a direct result of the assimilation process could therefore be considered as refinements to the world and be returned into the development loop requiring extension to the "establish world structure" process.

The criticisms and concerns of the Pentacle Method, noted during the evaluation process, indicate that the method is, by no means, the panacea for VRML world building. It was considered, however, that the application of the processes advocated by the method, to the identified scenario, did impart additional value to the overall world building process. This was based on the observation that

1. The method divides the process of world building into a clear series tasks providing ease of management.
2. The method provides a logical and sequential development route for these tasks via the primary development path.
3. The method is highly flexible by virtue of the lines of concern allowing timely information to be capitalised on through prototyping.

4. The provision for prototype experimentation by such lines affords developers the freedom to explore possible solutions in a manner that does not compromise the co-ordinating structure of the primary development path.

5. The method facilitates both the logical and physical structuring of a world and resources based around the client's requirements.

6. The method provides sufficient processes to facilitate quality assurance by auditable documentation trails, reviews and clearly defined deliverables.

7. The method identifies areas of specialisation for both the graphic artist and the software engineer auspices of project management and quality assurance devices.

It is therefore the conclusion of the evaluation process that the Pentacle Method is a tool that can be applied to bring structure, dynamism and additional value to the process of VRML world building. The evaluation process also concludes, however, that further work is required on the Pentacle Method if it is to be regarded as a *quality method* supporting the world building process.

6 Conclusions and Future Work.

"There has always been an unwritten rule that we only tackle problems that are truly understood and for which we can provide final solutions. Experimental extensions are encouraged, with the expectation that today's experimental extensions may be tomorrows VRML standard."

Gavin Bell. Silicon Graphics, Inc. (1996).

The project has identified the need for, shown the logical derivation from first principles of, and successfully implemented the Pentacle Method. Within each of the chapters a number of conclusions have been drawn in relation to the subject matter, it is therefore considered expedient to briefly review these in this the concluding chapter.

Chapter 1 presented an overview of the new Virtual Reality Modelling Language within its context as a new media for the dissemination of information using the Internets World Wide Web protocol. From the preliminary research conducted in support of the project this chapter concluded the two core hypotheses that formed the impetus for the project, namely,

1. The design requirements of VRML objects and their worlds are not well suited to existing design paradigms.
2. If VRML is to be a quality communication medium such a paradigm must exist.

Chapter 2 set out to establish the validity of these two hypotheses by researching those paradigms currently employed by the VRML world building community. The research conducted indicated that world builders were not using current software engineering paradigms. In response, a review three of the more popular software engineering paradigms was conducted in order to establish why such sound software engineering practices were clearly being ignored by the majority of the world building community. Based on the research and review of the

existing paradigms it was concluded that the apparent underuse of software engineering technique emitted from two root causes

1. The majority of VRML world builders are not software engineers but graphic artists with little or no formal software engineering training.
2. The existing paradigms reviewed, while each holding significance for world building, do not distinguish between the two distinct aspect of world building, the graphical content of and the assignment of behaviour to a world.

Chapter 3 explored the theoretical needs of a generic method and considered how these could be applied to the problem of VRML world building. The discussion examined potential strategies that could be employed by a structured method that would to redress the imbalance between the artistic and engineering needs of the world building process and allow the management and quality assurance aspects of a development project to be conducted in a relatively unobtrusive way. The chapter concluded that the concepts discussed provided a realistic model on which to base a method supporting all the considered aspects of the world building process.

Chapter 4 drew on the framework of the theoretical, method discussed in the previous chapter, in the attempt to present these concepts as a holistic method applicable to the management of a quality world building project.

Chapter 5 considered the implementation of the method to a case scenario in order to provide an assessment of the applicability to the VRML world building process. From the observations made during the creation of the scenario world, it was concluded that, while the method offered a degree of flexibility to the world building process, accommodating the two key areas of concern identified (the needs of the graphic artist and the software engineer), within a structure facilitating project management and quality assurance control, the method, as presented, raised a number of fundamental concerns.

6.1 Future Development of the Pentacle Method.

The initial observation of the project detailed in this paper, concludes that, if the Pentacle Method is to be a quality method supporting the VRML world building process, addressing the projects hypotheses, then a significant amount of future work must be undertaken.

6.1.1 Rectification of Identified Concerns.

Arguably, the priority for development of the Pentacle Method is an overhaul of the internal workings of the described point processes. It is considered that the initial starting point for this refinement will be based around the observations made during the evaluation of the method held in chapter 5.

6.1.2 Examination of Management Techniques.

Clearly the absence of the implementation of management techniques within the derivation of the case scenario using the Pentacle Method indicates a large area for development. It is proposed that the management techniques available to the project manager be reviewed in order to establish their applicability of use within the structure advocated by the method. It is considered that the ultimate goal of the review will not be to incorporate such techniques into the method, for such an act would restrict the project manager, but to recommend existing techniques that are appropriate to, and compliment the native management within the project. This it is considered will provide both flexibility and guidance to the veteran and novice project manager alike.

6.1.3 Examination of Quality Assurance Process.

More contentiously, the absence of the quality assurance processes within the evaluation of the Pentacle Method require the examination of quality assurance techniques in order to establish whether the method can be assimilated into existing practices or whether a entirely new strategy will need to be considered. This observation is based on an examination of the two main quality assurance standards, the ANSI/IEEE 730-1984 and 983-1986 standards and the ISO 9000 series

standards are based around the existing software engineering paradigms that have been shown as inappropriate to development using the VRML medium.

6.1.4 Development of Supporting Software.

The Pentacle Method advocates the use of various software applications for the derivation of VRML worlds during both development under the primary development path and the lines of concern. The method also requires a number of processes to be conducted parallel to the physical world building process ensuring timely information, correct documentation and resource availability at the respective development point. The development of a single tool that encompasses both the generation of worlds and supports the management of the project is clearly desirous. Currently there are two parties interested in developing a CASE tool based around the method although preliminary development of such a software tool has been restricted due to the author's current commitments.

6.1.5 Use of the Pentacle Method.

The evaluation process of the Pentacle Method relied on a number of assumptions, based on the limitations of human resources to the project (i.e. the author). As the method has been developed for utilisation within a project environment, the logical development of the method must include some form of co-operative work with other world builders. It is proposed that this be conducted using two identified resources.

1. It is hoped that the initial interest from the museum and school of conservation and restoration approached for sample artefacts can be capitalised on to secure an object and possibly personnel on which to implement a evaluation project.
2. The possibility for conducting a evaluation project is currently under discussion with those respondents to the initial questionnaire conducted in chapter2.

While the latter is highly dependant on the level of commitment that can be afforded to the project by those involved the possibility of developing a functional VRML world across it's native medium, the Internet, is to say the least, both intriguing and exciting.

6.2 Conclusion.

From the development of the method during the project and detailed in this paper it is the author's considered opinion that the Pentacle Method, as presented, representing a significant improvement to the methods currently employed by the VRML world building community.

The derived method, however, flawed and as such represents a realistic first draught of the method. It is considered that the origin of these flaws is primarily within the mechanics of the method's point processes and not within the fundamental principles used to build the method, nor with the conceptual model employed by the method itself. The need for further work to refine the method is clear if the method itself is to be recognised as being a quality method for the construction of VRML worlds by the world building community.

It is author's sincere hope that the interest thus far generated by the project, both from world builders and world participants alike, will enable future work to be conducted to the Pentacle Method, and that as a result the quality of worlds and, ultimately, the access and dispersal of information across the World Wide Web will be enhanced by the VRML medium.

References.

Adams, D. (1991).

In: Virtual Reality. London. Secker and Warburg Limited, pp1.

Avgerou, C., Cornford, T. (1993)

Developing Information Systems: Concepts, Issues and Practice. London, Macmillan, pp 130-133.

Allison, I (1996)

Taken form **Information Technology Management** lecture notes, Nottingham Trent University.

Baker, D., Tribe, J., Wills, H., Wilson, P. (1994).

Practical Applications of Artificial Reality. Undergraduate paper, Nottingham Trent University.

Baker, D., (1996).

Methodologies. The New Software Crisis? Undergraduate paper, Nottingham Trent University.

Bell, G., Behlendorf, B., Pesce, M. (1994)

The Virtual Reality Modelling Language v1.0 specification. Available WWW [Online] http://vag.vrml.org/www-vrml/vrml.tech/vrml0-3.htm.

Berners-Lee (1990)

HTML Specification Available WWW [Online] http://www:w3.org/htpertext/WWW/Arena/tour/start.html

Boehm. B., Pendo. M. H., Stuckle, E.D., Williams, R.D., Pyster A.B. (1975)

A software development environment for improving productivity. **IEEE Computer** 17(6), pp 30-42.

Boehm B., (1988)

A Spiral Model for Software Development and Enhancement. **Computer**, vol 21 5 pp61-72.

Bischofberger , W., Pomberger, G. (1990)

Prototyping Oriented Software Development. S.L, Springer-Verlag.

Brooks, F. (1975)

The Mythical Man-Month. Wokingham, UK. Addison-Wesley Publishing Company, pp 116.

Bricken, B. (1990)

Virtual Worlds: No Interface to Design. Technical Report R-90-2, Seattle, Human Interface Technology Laboratory (HITL), Washington Technology Centre, University of Washington, pp3.

Cox, K., Walker, D. (1994)

User Interface Design. 2nd ed. London. Prentice Hall, pp 51-55, 61.

Fitzgerald, B., (1996)

Formalised Systems Development Methodologies: a critical perspective. **Information System Journal**, 6, Oxford, Blackwell Science Ltd, pp25-40.

Mans, T., Coleman, M. (1996)

Software Quality Assurance. 2nd ed. Basingstoke, Macmillan Press, pp38-39.

Minsky, M. (1986)

The Society of Mind. New York. Simon and Schuster.

Pressman, R. (1994)

Software Engineering : A Practitioners Approach. 3rd ed. European ed. Ince, D. London, McGraw-Hill International., pp 26-28.

Rand, A. (1979)

Introduction to Objectivist Epistemology. New York. New American Library.

Roehl, B. (1996)

In correspondence with author. Brohl@sunee.uwaterloo.ca.

Rose D. (1994)

Life in the Virtual Village. **In Cyberlife** Grass, J. ed. Indianapolis, Sams Publishing, pp135-136.

SGI (1996)

Project Workflow. Available WWW [Online] http://vrml.sgi.com/workflow.html, Silicon Graphics Inc.

Sutherland, I (1965)

The Ultimate Display. **Proceedings of the IFIP Congress 1965**, pp506-508.

Sommerville, I (1996).

Software Engineering. 5th ed. Wokingham, UK. Addison-Wesley Publishing Company, pp 8-11.

Vacca, J. (1996)

VRML. Bringing Virtual Reality to the Internet. London. AP Professional, pp 29.

VAG (1996)

VRML 2 Specification. Available WWW [Online] http//vag.vrml.org/www-vrml/vrm.tech/vrml2.htm

Wastell, D. (1996)

Methodology as a Social Defence. **Information System Journal**, 6, Oxford, Blackwell Science Ltd, pp25-40.

Bibliography.

Ames. A, Nadeau. D, Moreland, J (1997)

VRML 2.0 Sourcebook. 2nd ed. Chichester, John Wiley & Sons Inc.

Foley, J, et al (1994)

Introduction to Computer Graphics. Harlow Addison Wesley

Gibson, W. (1984)

Neuromancer. New York, Berkley Publications Group.

Hartman, J. Wernecke, J (1996)

The VRML 2.0 Handbook: Building Moving Worlds on the Web. Silicon Graphics Inc. Harlow Addison Wesley Developers Press.

Matsuba, S. Roehl, B. (1996)

Using VRML Special Edition. Indianapolis, Que Corporation.

News Group (1996)

The Official VRML Newsgroup. Available News [Online] **news://comp.lang.vrml.**

Pressman, R. (1994)

Software Engineering : A Practitioners Approach. 3rd ed. European ed. Ince, D. London, McGraw-Hill International.

Rheingold, H. (1991)

Virtual Reality. London. Secker and Warburg Limited.

Stampe D, Roehl, B. Eagan, J. (1993)

Virtual Reality Creations: Explore, Manipulate and Create Virtual Worlds on Your PC. Corte Madera, California. The Waite Group Inc.

Williams, N. (1996)

Software Development Methodologies. Available WWW [Online] http://osiris.sund.ac.uk/~ca/nw1/html/swmintro.html.

SDSC (1996)

The VRML Repository. San Diego Supercomputer Centre. Available WWW [Online] http://www.sdsc.edu/wrml.

SGI (1996)

VRML Developers Forum. Silicon Graphics Inc. Available WWW [Online] http://vrml.sgi.com/overview.html.

Vacca, J. (1996)

VRML. Bringing Virtual Reality to the Internet. London. AP Professional.

Appendix A: Hierarchy Diagrams for ROMAN_POT.

Index to Inline Hierarchy Diagrams.

Diagram 1: ROMAN_POT Initial Primary Development Path.

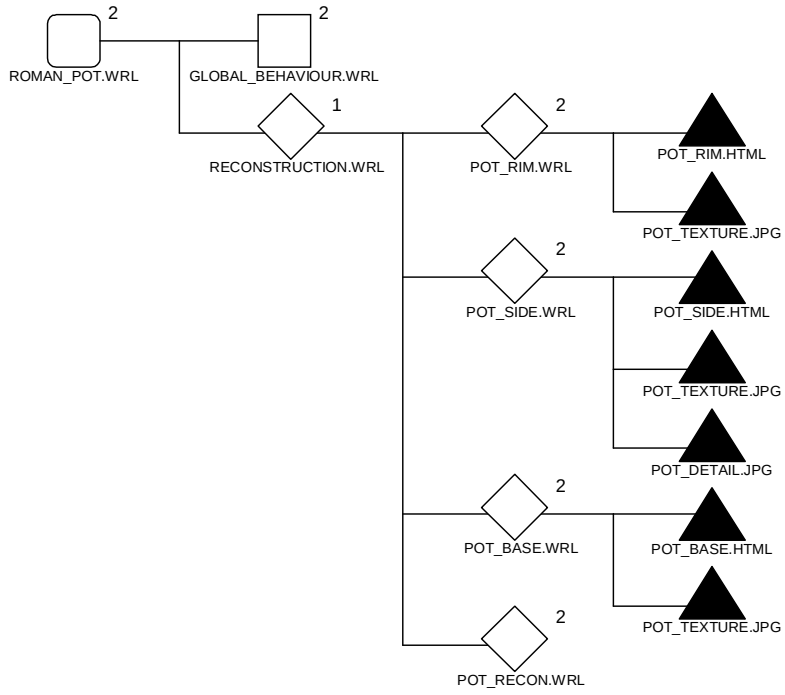


Diagram 2: Fragment Example (POT_SIDE) Initial Primary Development Path Sweep.

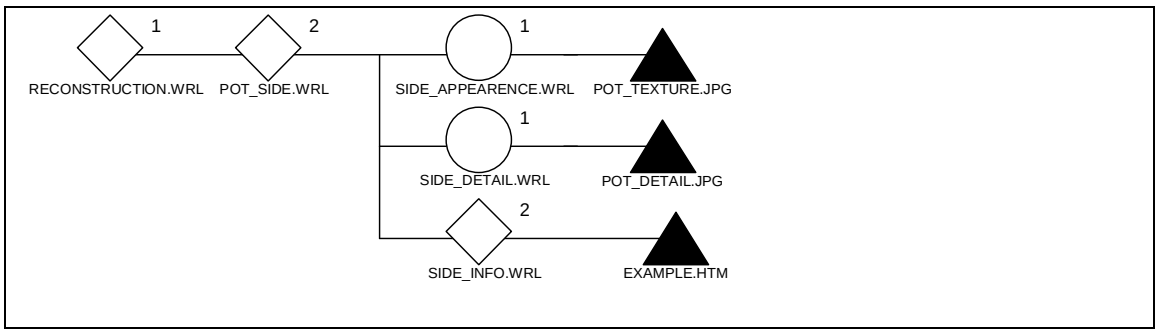

1
2
1
RECONSTRUCTION.WRL
POT_SIDE.WRL
SIDE_APPEARENCE.WRL
POT_TEXTURE.JPG
1
SIDE_DETAIL.WRL
POT_DETAIL.JPG
2
SIDE_INFO.WRL
EXAMPLE.HTM

Diagram 3: Example Information world (SIDE_INFO) Initial Primary Development Path.

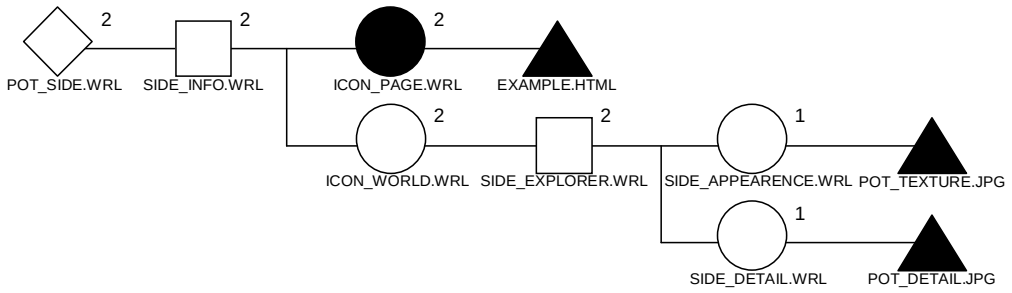


Diagram 4: Fragment Example (POT_SIDE) Cycle 2 Primary Development Path.

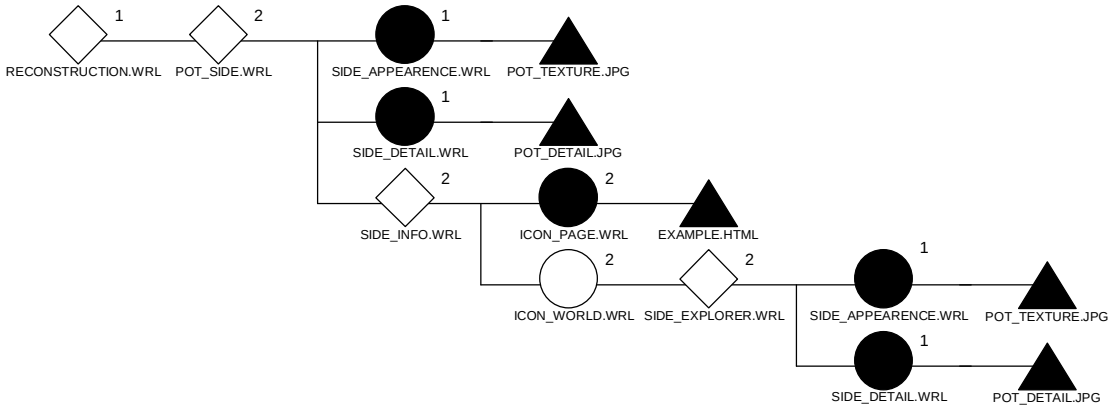


Diagram 5: Fragment Example (POT_SIDE) Cycle 3 Primary Development Path.

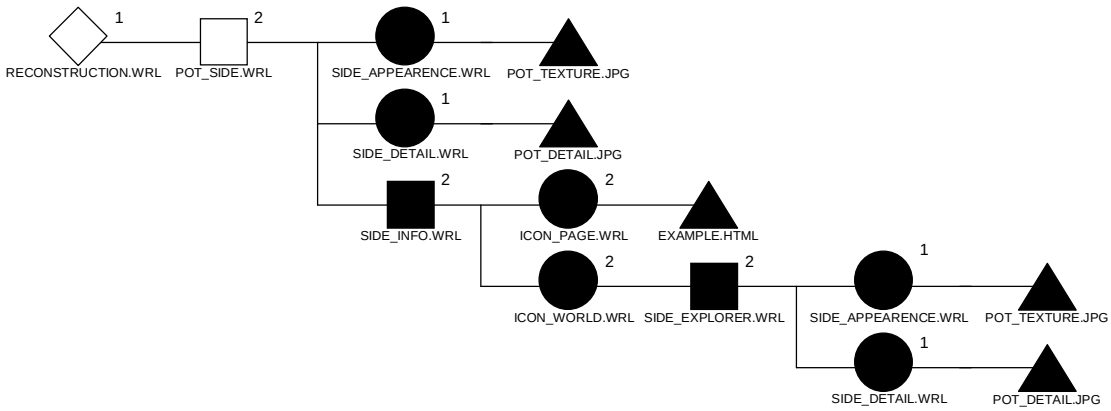


Diagram 6: Fragment Example (POT_SIDE) Cycle 4 Primary Development Path.

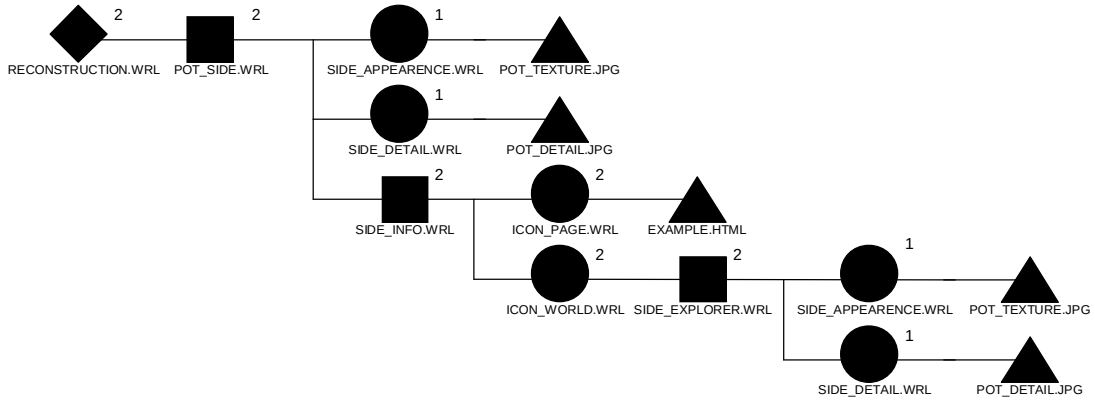


Diagram 7: GLOBAL_BEHAVIOR Initial Primary Development Path.

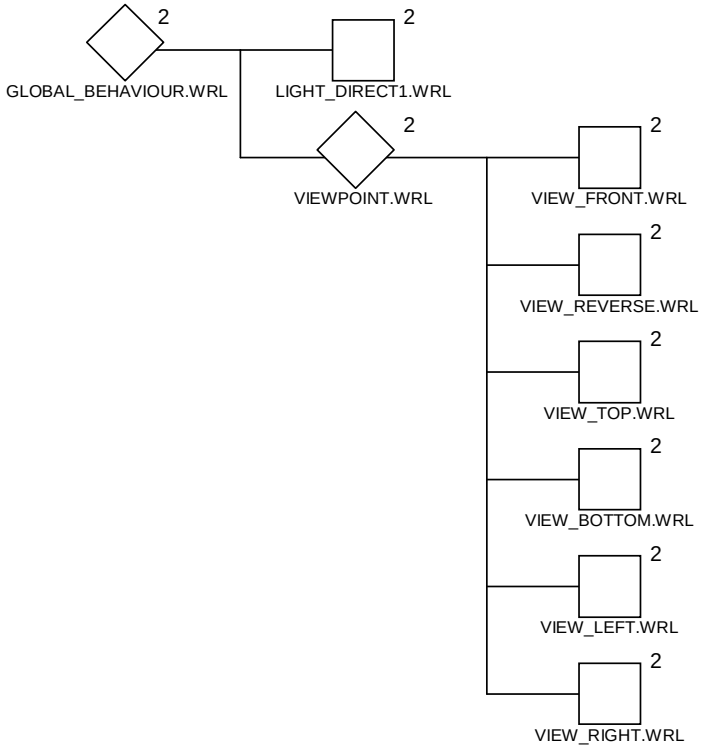


Diagram 8: GLOBAL_BEHAVIOR Cycle 2 Primary Development Path.

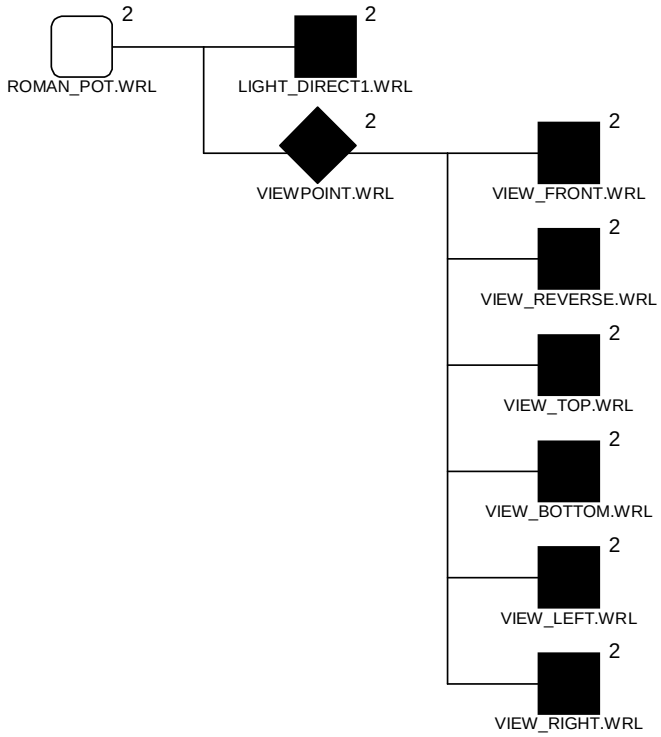
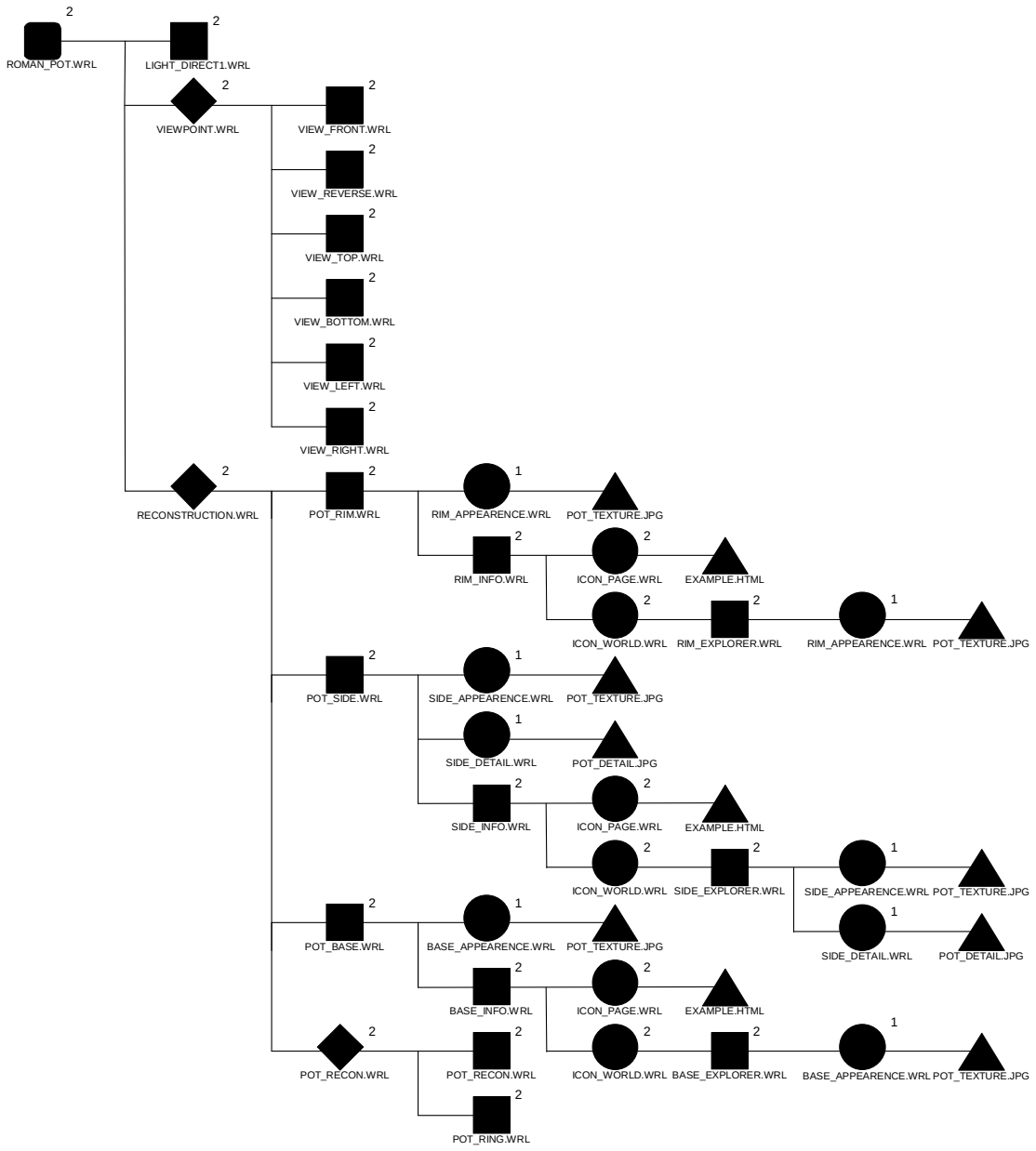


Diagram 9: ROMAN_POT Cycle 2 (Final) Primary Development Path.



Appendix B: VRML Code Listings.

Index To Code Section Listings.

1. The Universe World.

Code Section 1: ROMAN_POT.WRL

```
#VRML V2.0 utf8

Group
{
      children
      [

      Viewpoint
      {
            fieldOfView 0.75
            orientation 0 -1 0 3.11
            position 0 0 -150
      }
      Inline
      {
            url   "./common/lighting/light_direct1.wrl"
            bboxSize    0 0 0
            bboxCenter  0 0 0
      },
      Inline
      {
            url   "./common/viewpoint/viewpoint.wrl"
            bboxSize    0 0 0
            bboxCenter  0 0 0
      },
      Inline
      {
            url   "./reconstruction/reconstruction.wrl"
            bboxSize    0 0 0
            bboxCenter  0 0 0
      },
      ]
}
```

2. Global Effector (Universe Behaviour Node) Worlds.

Code Section 2:LIGHT_DIRECT1.WRL

```
#VRML V2.0 utf8

DirectionalLight
{
      intensity         0.5
      direction -0.2 -0.6 0.8
}
```

Code Section 3: VIEWPOINT.WRL

```
#VRML V2.0 utf8

Group
{
      children
      [
            Inline
            {
                  url   "view_forward.wrl"
                  bboxSize    0 0 0
                  bboxCenter  0 0 0
            },
            Inline
            {
                  url   "view_reverse.wrl"
                  bboxSize    0 0 0
                  bboxCenter  0 0 0
            },
            Inline
            {
                  url   "view_left.wrl"
                  bboxSize    0 0 0
                  bboxCenter  0 0 0
            },
            Inline
            {
                  url   "view_right.wrl"
                  bboxSize    0 0 0
                  bboxCenter  0 0 0
            },
            Inline
            {
                  url   "view_top.wrl"
                  bboxSize    0 0 0
                  bboxCenter  0 0 0
            },
            Inline
            {
                  url   "view_bottom.wrl"
                  bboxSize    0 0 0
                  bboxCenter  0 0 0
            },
      ]
}
```

Code Section 4: VIEW_BOTTOM.WRL

```
#VRML V2.0 utf8

Group
{
	children
	[
		Viewpoint
		{
			fieldOfView 0.75
			orientation 0.5 -0.5 0.5 2
			position 0 -150 0
			description "Bottom View"
		}
	]
}
```

Code Section 5: VIEW_FORWARD.WRL

```
#VRML V2.0 utf8

Group
{
	children
	[
		Viewpoint
		{
			fieldOfView 0.75
			orientation 0 -1 0 3.11
			position 0 0 -150
			description "Forward View"
		}
	]
}
```

Code Section 6: VIEW_LEFT.WRL

```
#VRML V2.0 utf8

Group
{
	children
	[
		Viewpoint
		{
			fieldOfView 0.75
			orientation 0 1 0 1.5
			position 150 0 0
			description "Left View"
		}
	]
}
```

Code Section 7:VIEW_REVERSE.WRL

```
#VRML V2.0 utf8

Group
{
	children
	[
		Viewpoint
		{
			fieldOfView 0.75
			orientation 0 0 1 0
			position 0 0 150
			description "Reverse View"
		}
	]
}
```

Code Section 8: VIEW_RIGHT.WRL

```
#VRML V2.0 utf8

Group
{
	children
	[
		Viewpoint
		{
			fieldOfView 0.75
			orientation 0 -1 0 1.5
			position -150 0 0
			description "Right View"
		}
	]

}
```

Code Section 9: VIEW_TOP.WRL

```
#VRML V2.0 utf8

Group
{
	children
	[
		Viewpoint
		{
			fieldOfView 0.75
			orientation -0.5 -0.5 -0.5 2
			position 0 150 0
			description "Top View"
		}
	]
}
```

Code Section 10: ICON_PAGE.WRL

```
#VRML V2.0 utf8
Group
{

      children Shape
      {
            appearance Appearance
            {
                  material Material
                  {
                        diffuseColor  1 0 0
                  }
            }
            geometry Box
            {
                  size  10 10 0.1  # field    SFVec3f
            }
      }
}
```

Code Section 11: ICON_WORLD.WRL

```
#VRML V2.0 utf8
Group
{

      children Shape
      {
            appearance Appearance
            {
                  material Material
                  {
                        diffuseColor  0 0 1
                  }
            }
            geometry Cylinder
            {
                  radius 5
                  height 0.1
            }
      }
}
```

4. Reconstruction (Universe Appearance Node) World.

Code Section 12: RECONSTRUCTION.WRL

```
#VRML V1.0 ascii

Separator
{
      WWWInline
      {
            name  "./pot_base/pot_base.wrl"
      }
      WWWInline
      {
            name  "./pot_side/pot_side.wrl"
      }
      WWWInline
      {
            name  "./pot_rim/pot_rim.wrl"
      }
      Group
      {
            Scale
            {
                  scaleFactor 1.72 1.7 1.72
            }
            WWWInline
            {
                  name  "./pot_recon/pot_recon.wrl"
            }
            WWWInline
            {
                  name  "./pot_recon/pot_ring.wrl"
            }
      }
}
```

5. Pot Reconstruction Worlds.

Code Section 13: POT_RECON.WRL

```
#VRML V2.0 utf8

Shape
{
      appearance Appearance
      {
            material Material
            {
                  diffuseColor 1 1 1
            }
      }
      geometry Extrusion
      {
            creaseAngle 1.57
            beginCap FALSE
            endCap   FALSE
            crossSection
            [
            0     -14
            -13   -16
            -18   -10
            -22   -4
            -22   7
            -21   10
            -20   11
            -18   13
            -18   18
            -20   21
            -21   22
            -23   21
            -25   18
            -23   16
            -25   13
            -26   10
            -27   0
            -25   -6
            -22   -12
            -17   -17
            -15   -22
            0     -19
            -13   -22
            ]
            spine
            [
            2     0     0.00,
            1.85  0     -0.77,
            1.41  0     -1.41,
            0.77  0     -1.85,
            0     0     -2.00,
            -0.77 0     -1.85,
            -1.41 0     -1.41,
            -1.85 0     -0.77,
            -2    0     0.00,
            -1.85 0     0.77,
            -1.41 0     1.41,
            -0.77 0     1.85,
```

```
            0     0     2.00,
            0.77  0     1.85,
            1.41  0     1.41,
            1.85  0     0.77,
            2     0     0.00
            ]
      }
}
```

Code Section 14: POT_RING.WRL

```
#VRML V2.0 utf8

Shape
{
      appearance Appearance
      {
            material Material
            {
                  diffuseColor 0.3 0.3 0.3
            }
      }
      geometry Extrusion
      {
            creaseAngle 1.57
            beginCap FALSE
            endCap   FALSE
            crossSection
            [
            -26   10
            -27   0
            ]
            spine
            [
            2     0     0.00,
            1.85  0     -0.77,
            1.41  0     -1.41,
            0.77  0     -1.85,
            0     0     -2.00,
            -0.77 0     -1.85,
            -1.41 0     -1.41,
            -1.85 0     -0.77,
            -2    0     0.00,
            -1.85 0     0.77,
            -1.41 0     1.41,
            -0.77 0     1.85,
            0     0     2.00,
            0.77  0     1.85,
            1.41  0     1.41,
            1.85  0     0.77,
            2     0     0.00
            ]
      }
}
```

6. Base Fragment Worlds

Code Section 15: POT_BASE.WRL

```
#VRML V2.0 utf8

Group
{
      children
      [
      DEF Pot_Base Transform
      {
            children
            [
            DEF Pot_Base_Group Inline
            {
                  url "./base_appearence.wrl"
            }
            DEF Pot_Base_Info Transform
            {
                  translation       0 0 60
                  #translation 0 0 0
                  children Inline
                  {
                        url "./base_info.wrl"
                  }
            }
            ]
      },
      DEF Sensor PlaneSensor { }
      ]
}
ROUTE Sensor.translation_changed TO Pot_Base.set_translation
```

Code Section 16:BASE_APPEARENCE.WRL

```
#VRML V1.0 ascii

Separator
{
      renderCulling AUTO

      DEF Pot_Base Separator
      {
            Material
            {
                  diffuseColor 0 0 1
            }
            Texture2
            {
                  filename "../../common/texture/pot_texture.jpg"
            }
            Coordinate3
            {
                  point
                  [
                  25.24 -6.98 25.24,
```

```
        28.53 -10.09 28.53,
        15.44 -10.09 37.27,
        13.66 -6.98 32.97,
        16.63 0.78 40.14,
        -15.44 -10.09 37.27,
        0 -10.09 40.35,
        0 -19.40 35.69,
        -13.66 -19.40 32.97,
        -16.63 0.78 40.14,
        0 0.78 43.45,
        15.44 -10.09 37.27,
        -10.68 -27.16 25.81,
        -8.91 -34.91 21.47,
        -7.72 -34.91 18.64,
        -15.44 -10.09 37.27,
        -13.66 -19.40 32.97,
        -16.63 0.78 40.14,
        -13.66 -6.98 32.97,
        -11.28 -16.29 27.24,
        -7.72 -25.60 18.64,
        0 -22.50 0,
        0 -30.26 0,
        16.63 0.78 40.14,
        0 -6.98 35.69,
        13.66 -19.40 32.97,
        25.24 -19.40 25.24,
        19.75 -27.16 19.75,
        10.69 -27.16 25.81,
        28.53 -10.09 28.53,
        11.28 -16.29 27.24,
        20.85 -16.29 20.85,
        25.24 -6.98 25.24,
        13.66 -6.98 32.97,
        7.72 -25.60 18.64,
        14.26 -25.60 14.26,
        18.64 -25.60 7.72,
        27.24 -16.29 11.28,
        -7.72 -34.91 18.64,
        0 -34.91 20.17,
        -8.91 -34.91 21.50,
        0 -34.91 23.28,
        -10.69 -27.16 25.81,
        0 -27.16 27.93,
        -11.28 -16.29 27.24,
        0 -16.29 29.48,
        -13.66 -6.98 32.97,
        -7.72 -25.60 18.64,
        0 -25.60 20.17,
        7.72 -34.91 18.64,
        8.91 -34.91 21.50,
        14.26 -34.91 14.26,
        16.46 -34.91 16.46,
        27.24 -16.29 11.28,
        25.19 -6.98 25.24,
        14.26 -34.91 14.26,
        16.46 -34.91 16.46
        ]
}
IndexedFaceSet
{
        coordIndex
        [
        2, 1, 0, -1,
```

```
2, 0, 3, -1,
2, 3, 4, -1,
7, 6, 5, -1,
7, 5, 8, -1,
6, 10, 9, -1,
6, 9, 5, -1,
11, 4, 10, -1,
11, 10, 6, -1,
14, 13, 12, -1,
12, 16, 15, -1,
12, 15, 17, -1,
12, 17, 18, -1,
12, 18, 19, -1,
12, 19, 20, -1,
12, 20, 21, -1,
12, 21, 22, -1,
12, 22, 14, -1,
10, 23, 3, -1,
10, 3, 24, -1,
10, 24, 18, -1,
10, 18, 17, -1,
27, 26, 25, -1,
27, 25, 28, -1,
26, 29, 11, -1,
26, 11, 25, -1,
32, 31, 30, -1,
32, 30, 33, -1,
31, 35, 34, -1,
31, 34, 30, -1,
37, 36, 35, -1,
37, 35, 31, -1,
22, 39, 38, -1,
39, 41, 40, -1,
39, 40, 38, -1,
41, 43, 42, -1,
41, 42, 40, -1,
43, 7, 8, -1,
43, 8, 42, -1,
24, 45, 44, -1,
24, 44, 46, -1,
45, 48, 47, -1,
45, 47, 44, -1,
48, 21, 47, -1,
22, 49, 39, -1,
49, 50, 41, -1,
49, 41, 39, -1,
50, 28, 43, -1,
50, 43, 41, -1,
28, 25, 7, -1,
28, 7, 43, -1,
25, 11, 6, -1,
25, 6, 7, -1,
33, 30, 45, -1,
33, 45, 24, -1,
30, 34, 48, -1,
30, 48, 45, -1,
34, 21, 48, -1,
22, 51, 49, -1,
51, 52, 50, -1,
51, 50, 49, -1,
52, 27, 28, -1,
52, 28, 50, -1,
35, 21, 34, -1,
```

```
                  36, 21, 35, -1,
                  37, 31, 0, -1,
                  29, 54, 53, -1,
                  36, 21, 22, -1,
                  36, 22, 55, -1,
                  36, 55, 56, -1,
                  36, 37, 1, -1,
                  36, 1, 26, -1,
                  36, 26, 27, -1,
                  36, 27, 52, -1
                  ]
            }
      }

}
```

Code Section 17: BASE_INFO.WRL

```
#VRML V2.0 utf8

Group
{
      children
      [
      Billboard
      {
            axisOfRotation  0 0 0
            children
            [
            DEF InfoWorld Anchor
            {
                  description     "Examine Base Fragment"
                  url             "base_explorer.wrl"
                  children
                  [
                  DEF Base_Icon_World Transform
                  {
                        translation -5 0 0
                        rotation    1.0 0.0 0.0 -1.57
                        children
                        [
                        Inline
                        {
                              url
      "../../common/navigation/icon_world.wrl"
                              bboxSize    0 0 0
                              bboxCenter  0 0 0
                        },
                        ]
                  }
                  ]
            },
            DEF InfoPage Anchor
            {
                  description     "Side Information Page"
                  url             "../../example_info.htm"
                  children
                  [
                  DEF Side_Icon_Page Transform
                  {
                        translation 5 0 0
                        children
```

```
                        [
                        Inline
                        {
                              url
      "../../common/navigation/icon_page.wrl"
                              bboxSize    0 0 0
                              bboxCenter  0 0 0
                        },
                        ]
                  }
                  ]
            }
            ]
      }
      ]
}
```

Code Section 18: BASE_EXPLORER.WRL

```
#VRML V2.0 utf8

NavigationInfo
{
      type              "EXAMINE"
}
Group {
      children
      [
            Viewpoint
            {
                  fieldOfView 0.75
                  orientation 0 0 1 0
                  position 0 0 150
                  description "Default View"
            }
            DEF Pot_Base_Group Transform
            {
                  children
                  [
                  DEF Pot_Base_World Inline
                  {
                        url "./base_appearence.wrl"
                  }
                  ]
            },
      DEF Touch TouchSensor { },
      DEF Clock TimeSensor
      {
            cycleInterval 10.0
      },
      DEF WorldPath OrientationInterpolator
      {
            key
            [
            0.0,
            0.11,
            0.22,
            0.33,
            0.44,
            0.55,
            0.66,
            0.77,
```

```
            0.88,
            1.0
            ]
            keyValue [
            0 0 0  0.0,
            1 0 0  1.1,
            1 0 0  2.2,
            1 0 0  3.3,
            0 1 0  4.4,
            0 1 0  5.5,
            0 1 0  6.6,
            0 0 1  7.7,
            0 0 1  8.8,
            0 0 1  9.9,
            ]
      }
      ]
}
ROUTE Touch.touchTime         TO Clock.set_startTime
ROUTE Clock.fraction_changed  TO WorldPath.set_fraction
ROUTE WorldPath.value_changed TO Pot_Base_Group.set_rotation
```

7. Side Fragment Worlds.

Code Section 19: POT_SIDE.WRL.

```
#VRML V2.0 utf8

Group
{
      children
      [
      DEF Pot_Side Transform
      {
            children
            [
            DEF Pot_Side_Group Transform
            {
                  children
                  [
                  DEF Pot_Side_World Inline
                  {
                        url "./side_appearence.wrl"
                  }
                  DEF Pot_Detail_World Inline
                  {
                        url "./side_detail.wrl"
                  },
                  ]
            }
            DEF Pot_Rim_Info Transform
            {
                  translation       -50 20 0
                  #translation 0 0 0
                  children Inline
                  {
                        url "./side_info.wrl"
                  }
            }
            ]
      },
      DEF Sensor PlaneSensor { }
      ]
}
ROUTE Sensor.translation_changed TO Pot_Side.set_translation
```

Code Section 20: SIDE_APPEARANCE.WRL

```
#VRML V1.0 ascii

Separator
{
      renderCulling AUTO

      Material
      {
            diffuseColor 0 1 0
      }
      Texture2
```

```
{
      filename "../../common/texture/pot_texture.jpg"
}
Coordinate3
{
      point
      [
      -40.35 -10.09 0,
      -37.27 -10.09 15.44,
      -32.97 -19.40 13.66,
      -35.69 -19.40 0,
      -43.45 0.78 0,
      -40.14 0.78 16.63,
      -38.79 20.95 0,
      -35.84 20.95 14.85,
      -38.71 16.29 16.03,
      -41.90 16.29 0,
      -31.03 17.85 0,
      -28.67 17.85 11.88,
      -27.24 20.95 11.28,
      -29.48 20.95 0,
      -34.14 16.29 0,
      -31.54 16.29 13.06,
      -35.69 11.64 0,
      -32.97 11.64 13.66,
      -35.69 -6.98 0,
      -32.97 -6.98 13.66,
      -29.48 -16.29 0,
      -27.24 -16.29 11.28,
      -30.72 0.78 30.72,
      -28.53 -10.09 28.53,
      -27.43 20.95 27.43,
      -29.63 16.29 29.63,
      -21.94 17.85 21.94,
      -20.85 20.95 20.85,
      -24.14 16.29 24.14,
      -25.24 11.64 25.24,
      -25.24 -6.98 25.24,
      -28.53 -10.09 -28.53,
      -37.27 -10.09 -15.44,
      -32.97 -19.40 -13.66,
      -25.24 -19.40 -25.24,
      -30.72 0.78 -30.72,
      -40.14 0.78 -16.63,
      -24.14 16.29 -24.14,
      -31.54 16.29 -13.06,
      -28.67 17.85 -11.88,
      -21.94 17.85 -21.94,
      -25.24 11.64 -25.24,
      -32.97 11.64 -13.66,
      -25.24 -6.98 -25.24,
      -32.97 -6.98 -13.66,
      -20.85 -16.29 -20.85,
      -27.24 -16.29 -11.28,
      -35.84 20.95 -14.85,
      -38.71 16.29 -16.03,
      -27.24 20.95 -11.28,
      -25.24 -6.98 25.24
      ]
}
IndexedFaceSet
{
      coordIndex
```

```
[
2, 1, 0, -1,
2, 0, 3, -1,
1, 5, 4, -1,
1, 4, 0, -1,
8, 7, 6, -1,
8, 6, 9, -1,
12, 11, 10, -1,
12, 10, 13, -1,
11, 15, 14, -1,
11, 14, 10, -1,
15, 17, 16, -1,
15, 16, 14, -1,
17, 19, 18, -1,
17, 18, 16, -1,
19, 21, 20, -1,
19, 20, 18, -1,
23, 22, 5, -1,
23, 5, 1, -1,
25, 24, 7, -1,
25, 7, 8, -1,
27, 26, 11, -1,
27, 11, 12, -1,
26, 28, 15, -1,
26, 15, 11, -1,
28, 29, 17, -1,
28, 17, 15, -1,
29, 30, 19, -1,
29, 19, 17, -1,
33, 32, 31, -1,
33, 31, 34, -1,
32, 36, 35, -1,
32, 35, 31, -1,
39, 38, 37, -1,
39, 37, 40, -1,
38, 42, 41, -1,
38, 41, 37, -1,
42, 44, 43, -1,
42, 43, 41, -1,
44, 46, 45, -1,
44, 45, 43, -1,
3, 0, 32, -1,
3, 32, 33, -1,
0, 4, 36, -1,
0, 36, 32, -1,
9, 6, 47, -1,
9, 47, 48, -1,
13, 10, 39, -1,
13, 39, 49, -1,
10, 14, 38, -1,
10, 38, 39, -1,
14, 16, 42, -1,
14, 42, 38, -1,
16, 18, 44, -1,
16, 44, 42, -1,
18, 20, 46, -1,
18, 46, 44, -1,
24, 25, 22, -1,
24, 22, 23, -1,
24, 23, 30, -1,
24, 30, 29, -1,
24, 29, 28, -1,
24, 28, 26, -1,
```

```
            24, 26, 27, -1,
            21, 30, 23, -1,
            19, 21, 50, -1,
            21, 23, 1, -1,
            21, 1, 2, -1,
            20, 21, 2, -1,
            20, 2, 3, -1,
            20, 3, 33, -1,
            20, 33, 34, -1,
            34, 45, 46, -1,
            34, 46, 20, -1,
            41, 48, 36, -1,
            41, 36, 35, -1,
            41, 35, 31, -1,
            31, 34, 45, -1,
            31, 45, 43, -1,
            31, 43, 41, -1,
            39, 40, 37, -1,
            37, 41, 48, -1,
            37, 48, 47, -1,
            47, 49, 39, -1,
            47, 39, 37, -1,
            13, 49, 47, -1,
            13, 47, 6, -1,
            13, 6, 7, -1,
            13, 7, 24, -1,
            24, 27, 12, -1,
            24, 12, 13, -1
            ]
      }
}
```

Code Section 21: SIDE DETAIL.WRL

```
#VRML V1.0 ascii

Separator
{
      renderCulling AUTO

      Material
      {
            diffuseColor 0 1 1
      }

      Texture2
      {
            filename "../../common/texture/pot_detail.jpg"
      }
      Coordinate3
      {
            point
            [
            -38.71 16.29 16.03,
            -29.63 16.29 29.63,
            -30.72 0.78 30.72,
            -40.14 0.78 16.63,
            -41.90 16.29 0,
            -43.45 0.78 0,
            -38.71 16.29 -16.03,
            -40.14 0.78 -16.63
            ]
```

```
	}
	IndexedFaceSet
	{
		coordIndex
		[
		2, 1, 0, -1,
		2, 0, 3, -1
		]
	}
	IndexedFaceSet
	{
		coordIndex
		[
		3, 0, 4, -1,
		3, 4, 5, -1
		]
	}
	IndexedFaceSet
	{
		coordIndex
		[
		5, 4, 6, -1,
		5, 6, 7, -1
		]
	}
}
```

Code Section 22: SIDE_INFO.WRL

```
#VRML V2.0 utf8

Group
{
	children
	[
		Billboard
		{
			axisOfRotation  0 0 0
			children
			[
			DEF InfoWorld Anchor
			{
					description     "Examine Side Fragment"
					url             "side_explorer.wrl"
					children
					[
					DEF Side_Icon_World Transform
					{
						translation -5 0 0
						rotation    1.0 0.0 0.0 -1.57
						children
						[
						Inline
						{
							url
	"../../common/navigation/icon_world.wrl"
							bboxSize    0 0 0
							bboxCenter  0 0 0
						},
						]
```

```
                              }
                              ]
                        },
                        DEF InfoPage Anchor
                        {
                              description     "Side Information Page"
                              url             "../../example_info.htm"
                              children
                              [
                              DEF Side_Icon_Page Transform
                              {
                                    translation 5 0 0
                                    children
                                    [
                                    Inline
                                    {
                                          url
      "../../common/navigation/icon_page.wrl"
                                          bboxSize    0 0 0
                                          bboxCenter  0 0 0
                                    },
                                    ]
                              }
                              ]
                        }
                  ]
            }
      ]
}
```

Code Section 23: SIDE_EXPLORER.WRL

```
#VRML V2.0 utf8

NavigationInfo
{
      type             "EXAMINE"
}
Group {
      children
      [
            Viewpoint
            {
                  fieldOfView 0.75
                  orientation 0 -1 0 1.5
                  position -150 0 0
                  description "Default View"
            }
            DEF Pot_Side_Group Transform
            {
                  children
                  [
                  DEF Pot_Side_World Inline
                  {
                        url "./side_appearence.wrl"
                  }
                  DEF Pot_Detail_World Inline
                  {
                        url "./side_detail.wrl"
                  },
                  ]
```

```
            },
      DEF Touch TouchSensor { },
      DEF Clock TimeSensor
      {
            cycleInterval 10.0
      },
      DEF WorldPath OrientationInterpolator
      {
            key
            [
            0.0,
            0.11,
            0.22,
            0.33,
            0.44,
            0.55,
            0.66,
            0.77,
            0.88,
            1.0
            ]
            keyValue [
            0 0 0  0.0,
            1 0 0  1.1,
            1 0 0  2.2,
            1 0 0  3.3,
            0 1 0  4.4,
            0 1 0  5.5,
            0 1 0  6.6,
            0 0 1  7.7,
            0 0 1  8.8,
            0 0 1  9.9,
            ]
      }
      ]
}
ROUTE Touch.touchTime         TO Clock.set_startTime
ROUTE Clock.fraction_changed  TO WorldPath.set_fraction
ROUTE WorldPath.value_changed TO Pot_Side_Group.set_rotation
```

8. Rim Fragment Worlds.

Code Section 24: POT_RIM.WRL

```
#VRML V2.0 utf8

Group
{
      children
      [
      DEF Pot_Rim Transform
      {
            children
            [
            DEF Pot_Rim_Group Inline
            {
                              url "./rim_appearence.wrl"
            }
            DEF Pot_Rim_Info Transform
            {
                  translation       50 40 20
                  #translation 0 0 0
                  children Inline
                  {
                        url "./rim_info.wrl"
                  }
            }
            ]
      },
      DEF Sensor PlaneSensor { }
      ]
}
ROUTE Sensor.translation_changed TO Pot_Rim.set_translation
```

Code Section 25: RIM_APPEARANCE.WRL

```
#VRML V1.0 ascii

Separator
{
      renderCulling AUTO

      DEF Pot_Rim Separator
      {
            Material
            {
                  diffuseColor 1 0 0
            }
            Texture2
            {
                  filename "../../common/texture/pot_texture.jpg"
            }
            Coordinate3
            {
                  point
                  [
                  14.25 25.60 34.41,
```

```
      26.33 25.60 26.33,
      27.43 20.95 27.43,
      14.85 20.95 35.84,
      14.85 28.71 35.84,
      27.43 28.71 27.43,
      11.28 20.95 27.24,
      20.85 20.95 20.85,
      20.85 28.71 20.85,
      11.28 28.71 27.24,
      34.41 25.60 14.25,
      35.84 20.95 14.85,
      35.84 28.71 14.85,
      26.33 33.36 26.33,
      34.41 33.36 14.25,
      24.14 34.91 24.14,
      31.54 34.91 13.06,
      21.94 33.36 21.94,
      28.67 33.36 11.88,
      27.24 28.71 11.28,
      27.24 20.95 11.28,
      37.24 25.60 0,
      38.79 20.95 0,
      38.79 28.71 0,
      37.24 33.36 0,
      34.14 34.91 0,
      31.03 33.36 0,
      29.48 28.71 0,
      29.48 20.95 0,
      34.41 25.60 -14.25,
      35.84 20.95 -14.85,
      35.84 28.71 -14.85,
      34.41 33.36 -14.25,
      31.54 34.91 -13.06,
      28.67 33.36 -11.88,
      27.24 28.71 -11.28,
      27.24 20.95 -11.28,
      26.33 33.36 -26.33,
      27.43 28.71 -27.43,
      24.14 34.91 -24.14,
      21.94 33.36 -21.94,
      20.85 28.71 -20.85,
      20.85 20.95 -20.85
      ]
}

IndexedFaceSet
{
      coordIndex
      [
      2, 1, 0, -1,
      2, 0, 3, -1,
      1, 5, 4, -1,
      1, 4, 0, -1,
      8, 7, 6, -1,
      8, 6, 9, -1,
      11, 10, 1, -1,
      11, 1, 2, -1,
      10, 12, 5, -1,
      10, 5, 1, -1,
      12, 14, 13, -1,
      12, 13, 5, -1,
      14, 16, 15, -1,
      14, 15, 13, -1,
```

```
16, 18, 17, -1,
16, 17, 15, -1,
18, 19, 8, -1,
18, 8, 17, -1,
19, 20, 7, -1,
19, 7, 8, -1,
22, 21, 10, -1,
22, 10, 11, -1,
21, 23, 12, -1,
21, 12, 10, -1,
23, 24, 14, -1,
23, 14, 12, -1,
24, 25, 16, -1,
24, 16, 14, -1,
25, 26, 18, -1,
25, 18, 16, -1,
26, 27, 19, -1,
26, 19, 18, -1,
27, 28, 20, -1,
27, 20, 19, -1,
30, 29, 21, -1,
30, 21, 22, -1,
29, 31, 23, -1,
29, 23, 21, -1,
31, 32, 24, -1,
31, 24, 23, -1,
32, 33, 25, -1,
32, 25, 24, -1,
33, 34, 26, -1,
33, 26, 25, -1,
34, 35, 27, -1,
34, 27, 26, -1,
35, 36, 28, -1,
35, 28, 27, -1,
38, 37, 32, -1,
38, 32, 31, -1,
37, 39, 33, -1,
37, 33, 32, -1,
39, 40, 34, -1,
39, 34, 33, -1,
40, 41, 35, -1,
40, 35, 34, -1,
41, 42, 36, -1,
41, 36, 35, -1,
29, 30, 38, -1,
30, 42, 38, -1,
38, 42, 41, -1,
38, 41, 40, -1,
38, 40, 39, -1,
38, 39, 37, -1,
38, 31, 29, -1,
9, 15, 17, -1,
9, 17, 8, -1,
15, 4, 5, -1,
15, 5, 13, -1,
9, 4, 15, -1,
6, 3, 0, -1,
6, 0, 4, -1,
6, 4, 9, -1,
2, 11, 22, -1,
2, 22, 30, -1,
30, 42, 36, -1,
30, 36, 28, -1,
```

```
                  30, 28, 20, -1,
                  7, 6, 3, -1,
                  7, 3, 2, -1,
                  2, 30, 20, -1,
                  2, 20, 7, -1
                  ]
            }
      }

}
```

Code Section 26: RIM_INFO.WRL

```
#VRML V2.0 utf8

Group
{
      children
      [
      Billboard
      {
            axisOfRotation  0 0 0
            children
            [
            DEF InfoWorld Anchor
            {
                  description     "Examine Rim Fragment"
                  url             "rim_explorer.wrl"
                  children
                  [
                  DEF Side_Icon_World Transform
                  {
                        translation -5 0 0
                        rotation    1.0 0.0 0.0 -1.57
                        children
                        [
                        Inline
                        {
                              url
      "../../common/navigation/icon_world.wrl"
                              bboxSize    0 0 0
                              bboxCenter  0 0 0
                        },
                        ]
                  }
                  ]
            },
            DEF InfoPage Anchor
            {
                  description     "Side Information Page"
                  url             "../../example_info.htm"
                  children
                  [
                  DEF Side_Icon_Page Transform
                  {
                        translation 5 0 0
                        children
                        [
                        Inline
                        {
```

```
                              url
      "../../common/navigation/icon_page.wrl"
                              bboxSize    0 0 0
                              bboxCenter  0 0 0
                        },
                        ]
                  }
                  ]
            }
            ]
      }
      ]
}
```

Code Section 27: RIM_EXPLORER.WRL

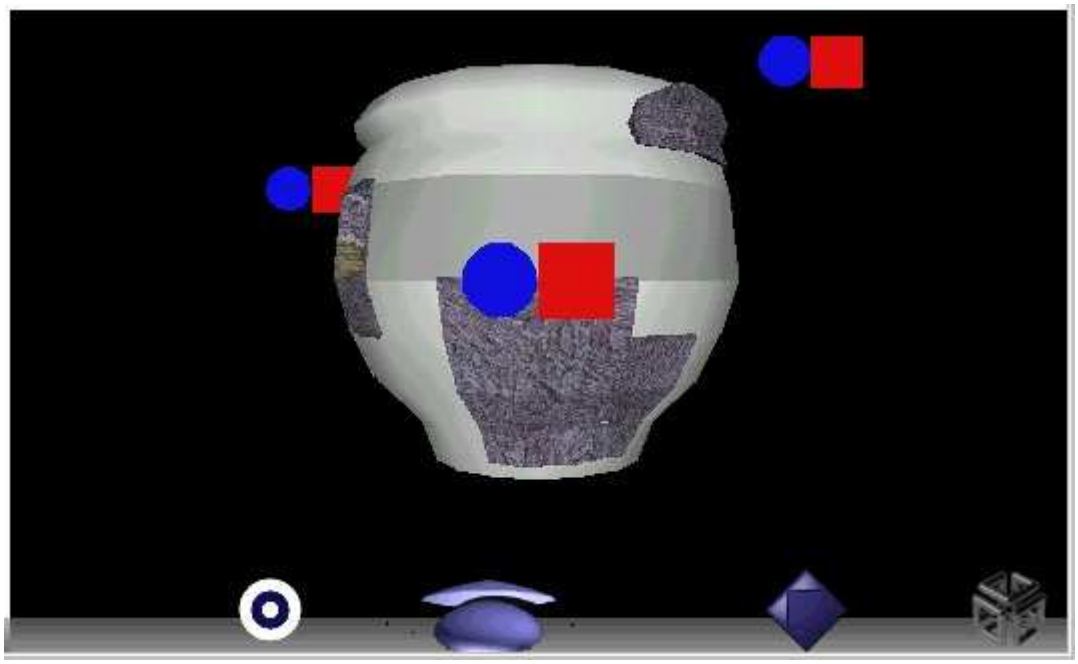
```
#VRML V2.0 utf8

NavigationInfo
{
      type              "EXAMINE"
}
Group {
      children
      [
            Viewpoint
            {
                  fieldOfView 0.75
                  orientation 0 1 0 1.5
                  position 150 0 0
                  description "Default View"
            }
            DEF Pot_Rim_Group Transform
            {
                  children
                  [
                  DEF Pot_Rim_World Inline
                  {
                        url "./rim_appearence.wrl"
                  }
                  ]
            },
      DEF Touch TouchSensor { },
      DEF Clock TimeSensor
      {
            cycleInterval 10.0
      },
      DEF WorldPath OrientationInterpolator
      {
            key
            [
            0.0,
            0.11,
            0.22,
            0.33,
            0.44,
            0.55,
            0.66,
            0.77,
            0.88,
            1.0
```

```
            ]
            keyValue [
            0 0 0  0.0,
            1 0 0  1.1,
            1 0 0  2.2,
            1 0 0  3.3,
            0 1 0  4.4,
            0 1 0  5.5,
            0 1 0  6.6,
            0 0 1  7.7,
            0 0 1  8.8,
            0 0 1  9.9,
            ]
      }
      ]
}
ROUTE Touch.touchTime         TO Clock.set_startTime
ROUTE Clock.fraction_changed  TO WorldPath.set_fraction
ROUTE WorldPath.value_changed TO Pot_Rim_Group.set_rotation
```

Appendix C: Screen Shots of ROMAN_POT.WRL.

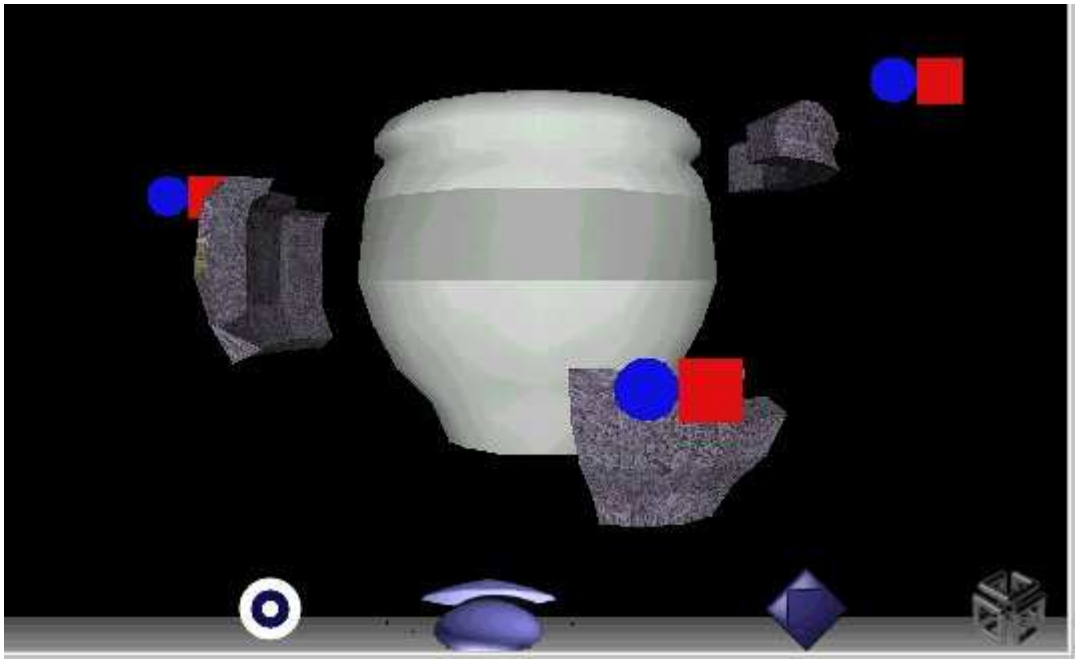
Index to Screen Shots.

Screen Shot 1: Roman Pot World (Reverse View).



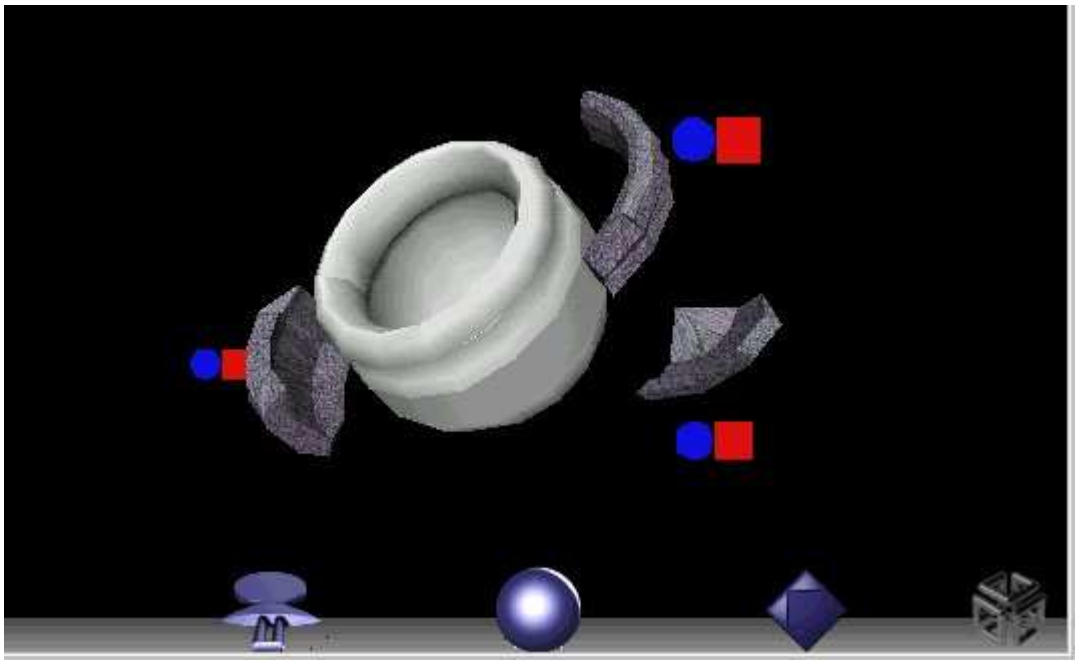
NB: Using Silicon Graphic's Inc CosmoPlayer Plug-in for MS Internet Explorer 95.

Screen Shot 2: Roman Pot World (Exploded View).



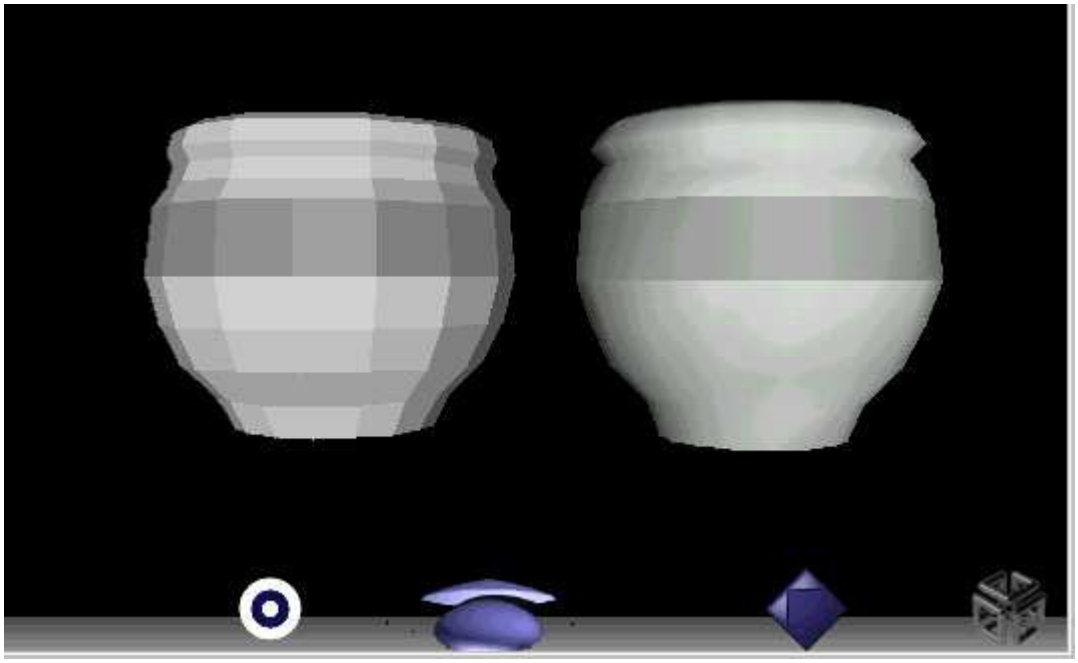
NB: Expanded using click and drag behaviour.

Screen Shot 3: Roman Pot World (Billboard Detail).



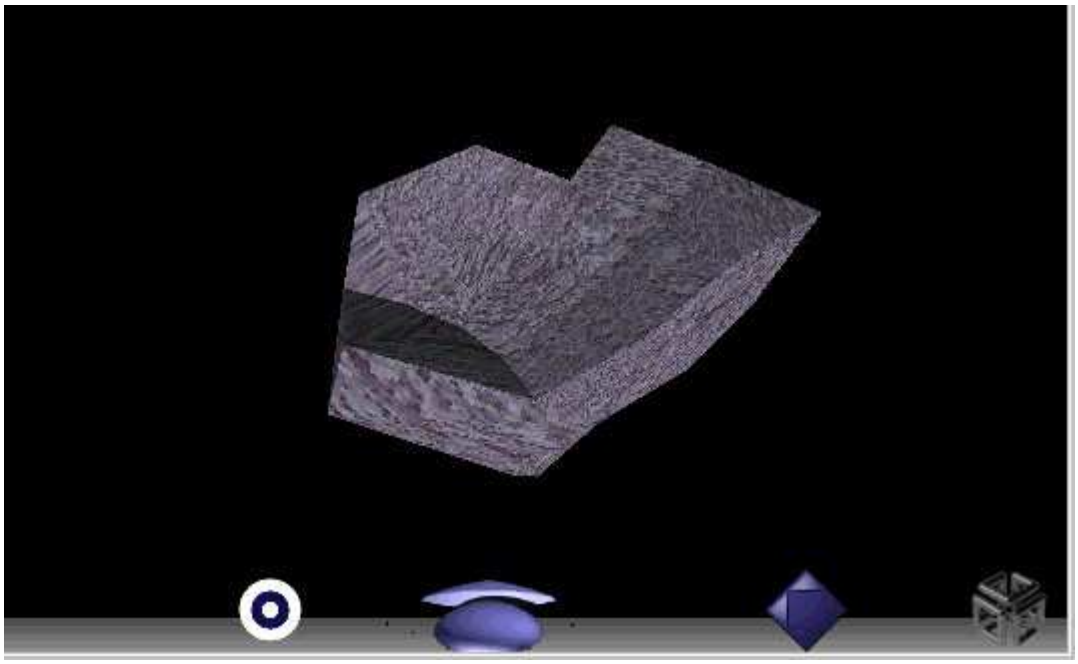
NB: Rotation using examiner tool does not effect the billboard icon group's attitude.

Screen Shot 4: Reconstruction World Before and After Optimisation.



NB: Left VRML 1 version using point set, Right VRML 2 version using double extrusion.

Screen Shot 5: Base Fragment World.

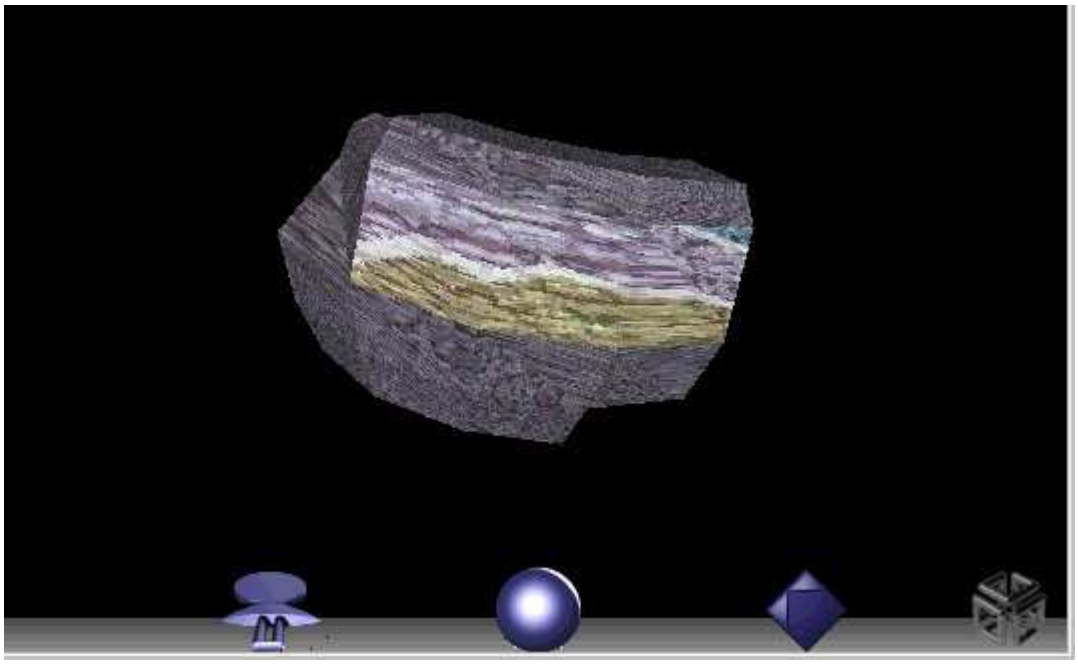


NB: Rotated using click behaviour.

Screen Shot 6: Rim Fragment World.

NB: Positioned freehand using native browser Walk and Examine functionality.

Screen Shot 7: Side Fragment World.



NB: Note the error in the detail texture mapping, a known problem with CosmoPlayer.