

**University of Derby – School of Computing**

**A project completed as part of the requirements for the  
BSc (Hons) Computer Games Programming**

**entitled**

**Investigating a model of place for game AI and evaluating its performance as a method for  
authoring agent reactions**

**by**

**Henry H. C. Golding**

**in the years 2009 - 2010**

## Abstract

This project examines whether concepts of place can be usefully applied in game AI, specifically in the context of authoring agent reactions. It reviews current spatial approaches to world representation in games and literature on place from other disciplines. A conceptual model of place for use in game AI is then proposed, influenced by the theories of implacement and affordances. A place agnostic test case is presented, together with an analysis of two parallel implementations of it; a control implementation and an implementation derived from the proposed model of place. The implementations are then analyzed in objective terms, revealing that the place-based implementation performs worse but supports authoring better. The recommendation of this report is to use place-based representations judiciously, bearing in mind their additional overheads as well as their authoring benefits.

## **Acknowledgements**

I would like to thank my supervisor, Adam Russell, for his unfailing interest in the project and for giving so generously of his time throughout the process, as well as originally inspiring this project.

I would also like to thank John Sear for his exemplary leadership throughout my time at Derby, and his consistent focus on technical excellence.

This project would not have been possible without the love and support of my family, both in Derby and elsewhere. Thank you.

## Table of Contents

Abstract .....	2
Acknowledgements.....	3
Table of Figures .....	7
1. Introduction .....	8
1.1. Aims and Objectives.....	8
1.2. Methodology.....	9
2. Literature Review .....	10
2.1. Spatial Approaches Found in Game AI.....	10
2.2. Why Place? .....	11
2.3. Applications of Place-Based Study .....	11
2.4. What is Place? .....	13
2.5. Postmodern Placelessness .....	15
2.6. Affordances .....	16
2.7. Existing Models of Place.....	17
3. Research.....	20
3.1. A Model of Place for Game AI .....	20
3.1.1. An Overview of the Proposed Model.....	20
3.1.2. Agent, Implacer and Body .....	21
3.1.3. Space – the Physical Characteristics of the Place.....	22
3.1.4. Place-world and Place Hierarchy.....	22
3.1.5. Affordances .....	23
3.1.6. Measures of Implacement .....	24
3.2. Detailed Methodology .....	25
3.2.1. Test Case/Behaviour Specification.....	25
3.2.2. Extension to the Test Case .....	25
3.2.3. Evaluation.....	26

3.3.	Framework Implementation .....	26
3.3.1.	Spatial Representation .....	28
3.3.2.	Action Queries .....	29
3.3.3.	Reaction Queries .....	29
3.4.	Control Implementation.....	30
3.4.1.	Spatial Representation .....	30
3.4.2.	Action Queries.....	31
3.4.3.	Reaction Queries .....	31
3.4.4.	Extension Implementation.....	31
3.4.5.	Authoring.....	31
3.5.	Place-Based Implementation .....	32
3.5.1.	Overview .....	32
3.5.2.	Spatial Representation .....	34
3.5.3.	Action Queries.....	34
3.5.4.	Reaction Queries .....	34
3.5.5.	Extension Implementation .....	35
3.5.6.	Authoring.....	35
4.	Results .....	36
4.1.	Description of Objective Metrics.....	36
4.2.	Objective Metrics .....	37
4.3.	Comparison .....	38
4.3.1.	Structure.....	38
4.3.2.	Complexity.....	39
4.3.3.	Memory.....	40
4.3.4.	Performance.....	40
4.4.	Projections Based on Extension Data.....	41
5.	Conclusions and Recommendations .....	43
5.1.	Criticism of Methodology and Future Directions.....	44

Bibliography .....	46
Critical Evaluation.....	49
Appendix 1 – Project Proposal Form.....	51
Appendix 2 – Project Progress Sheets.....	51
Appendix 3 – Project Plan Gantt Chart .....	52
Appendix 4 – Behaviour Specification and Extension Notes.....	53
Appendix 5 – CCCC Analysis Data.....	57
Appendix 6 – Control Implementation Diagram .....	58
Appendix 7 – Place-Based Implementation Diagram.....	59

## Table of Figures

Figure 1: One possible Why, What, How level description of a Restaurant (from Jordan et al., 1998)	18
Figure 2: A physical environment and its causal/topological map. The map has 20 distinctive states and 7 places (image from Kuipers & Beeson, 2002)	19
Figure 3: Overview of the proposed model of place	21
Figure 4: An analogy for an 'implacer', implaced in multiple places	21
Figure 5: An example of hierarchy and layering	23
Figure 6: The client application's interface to the behaviour system	27
Figure 7: The enumerations used in the framework	28
Figure 8: The framework/façade's API, as presented to the client application	28
Figure 9: The design of the framework and interfaces to the implementations	30
Figure 10: Objective metrics	37
Figure 11: Code breakdown - proportional	38
Figure 12: Code breakdown - quantitative	38
Figure 13: Comparison of complexity measures	39
Figure 14: Complexity measures apportioned to test case specific code	39
Figure 15: Memory usage – comparison	40
Figure 16: Performance differences between control and place-based implementations	40
Figure 17: Projected code growth plotted against number of extensions	41
Figure 18: Projected total codebase size as number of extensions grow	42
Figure 19: Guidance for suitable applications of place-based models	44

## 1. Introduction

This project sets out to investigate the advantages and disadvantages of using a model of place in game AI to facilitate the authoring of NPC (Non-Player Character) reactions (e.g. to player character actions, other agents' actions, or world events).

The term 'space' usually refers to absolute positions or regions in some defined (e.g. Cartesian) system. 'Place', on the other hand, is a more nebulous term concerned with layered, subjective meanings or semantics mapped onto physical 'space'; while 'space' deals with the objective physical environment, 'place' deals with the subjective, perceived, environment of an agent.

Game AI programmers are often involved in the work of implementing artificial agents in rich virtual environments. As environments become more complex there has been interest in increasing the spatial competence of agents (Isla, 2005) and it has been proposed that concepts of place may be a means of accomplishing this (Russell, 2008).

This section of the project sets out its aims and methodology, while section 2 reviews the available literature on concepts of place from diverse fields.

Section 3 proposes a conceptual model of place for game AI, sets out a test case and describes two parallel implementations of the test case. The results of this research are examined in section 4, while section 5 draws conclusions from these findings.

### 1.1. Aims and Objectives

This project is motivated by Adam Russell's suggestion (Russell, 2008) that place-based game world models could be a means of adding meaning and richness to the worlds of AI agents, leading to more believable behaviour (or, more specifically, helping to avoid unbelievable behaviours). The overarching aim of this project, then, is to investigate Russell's suggestion. It seeks to answer the question "Do implementations based on concepts of place provide any measurable benefits in game AI".

The project does not focus on believability benefits (the experience of the end user), but instead on implementation and authoring benefits. Believability is subjective and hard to measure, and similar levels of believability and complexity of behaviour could be achieved through a variety of different methods. It is the promise of making believable agents easier to author and implement that is the key reason for pursuing this line of enquiry.

The area of authoring agent reactions was chosen as one of many possible applications of a place-based world model, as it offers the possibility of both subjective (by content creators) and objective

(by comparison to a reference implementation) analysis. The area is of importance to the computer games industry because AI agents displaying inappropriate reactions are very conspicuous failures in a game's AI.

Agent reactions were also expected to be a particularly suitable test case for a place-based system due to their highly place dependent nature. 'Out of place' actions nearly always provoke a particular kind of social response, while the same action 'in place' may not provoke any reaction at all. Further, the same action may have different meanings in different places. For example, consider the reaction of an ordinary citizen witnessing a gun being fired in the street, compared to the reaction if the gun were fired on a shooting range, or in a battle. The reaction would differ depending on where the action *took place*.

However, place is not the only influence on an agent's reaction. Other factors, such as the agent's own personality, also contribute to a believable reaction (Shaw et al., 2001).

## **1.2. Methodology**

The methodology adopted comprised a number of key stages (refer to appendix 3 for a Gantt chart showing the original project plan).

First, the available literature on place and place-based systems of representation was reviewed. Then a model of place for use in game AI, informed by the results of the literature review, was synthesized.

A place agnostic behaviour specification was then constructed (section 3.2) comprising a number of different factions, a known set of stimuli and the desired reactions to them in particular locations. This specification was used as a test case for two parallel implementations of a behaviour library; a control implementation and an implementation employing the proposed model of place. The two implementations were then compared and evaluated against objective criteria.

---

## 2. Literature Review

The guiding principle of this literature review was to acquire and analyze research to inform the design of a useful model of place for game AI. Sources consulted during the literature review included books, conference proceedings, journal articles and internet sources.

### 2.1. Spatial Approaches Found in Game AI

Abstract spatial representations have long been used in game AI for the purposes of movement and pathfinding, with the majority of modern games using navigation meshes and variations of the A\* graph search algorithm (Millington & Funge, 2009).

The influence maps proposed by Tozour (2001), were a first step towards combining physical spatial data with intangible semantic data about the simulated world. The concept has been used by others to improve the believability of pathfinding methods (e.g. the 'risk averse' pathfinding proposed by Paanakker (2008) based on influence maps) and in combat AI to calculate cover points (Straatman et al., 2006).

There is also increasing interest in the dynamic generation of navigation meshes, including strategic information, to account for modern game environments where the spatial environment is continually reconfigured at runtime. Axelrod (2008) proposes the *render-generate* algorithm; an efficient method of updating navigation graphs at runtime in highly dynamic environments. He considers semantic concepts such as 'walkability' and proposes future research into the use of influence maps to avoid "problematic places". He also considers the ability of agents to reconfigure the spatial environment themselves, for example by opening a door, or destroying a wall. Meanwhile, Olsson (2008) discusses another approach to terrain analysis that extracts semantic data from a purely spatial world representation, e.g. choke-points, 'game-zones' and 'hotspots'.

The apparent trend in game AI from simple pathfinding towards agents with richer, contextual, understandings of their environments is most clearly evidenced by Damián Isla's 2005 presentation "*Dude: Where's my Warthog? From Pathfinding to General Spatial Competence*", discussing how to achieve the appearance of spatial competence in agents. Listing over 40 individual competencies that were implemented in Halo2, he suggests that an agent is still unable to reason about space believably without having a more complex environmental representation. He highlights the difference between geometric convenience (a path can be found) and conceptual convenience (the path 'makes sense'), and proposes the use of concepts of place as a solution to this problem.

It appears, then, that there is a general movement in game AI towards more complex world representations than have traditionally been used for pathfinding; combining spatial and semantic data to improve agents' decision making.

## 2.2. Why Place?

Interest was found in concepts of place in diverse fields, including philosophy, cultural geography, cartography, geographic information science, cognitive psychology, game AI, robotics and human computer interaction. In general, each field refers to different source material when investigating concepts of place, with little evidence of cross-pollination. However, many sources encountered during the review (across diverse fields) show remarkably similar attitudes to place.

The concept of place is important to most of the sources encountered and, further, they often believe that place is in danger of being overlooked. In philosophy, Edward Casey rails against the subordination of 'place' to science's 'time' and 'space' (Casey, 1993), while in game AI, Adam Russell laments the "impoverished *umwelt*" (subjective environment) of virtual agents and their lack of a sense of place (Russell, 2008).

Casey's own work offers a possible explanation for this interest in place. He argues that place is of prime importance to the human observer; that a world without place is inconceivable and utterly terrifying, that without a 'here' we are nowhere and nothing. Given this presumed bias, it may be especially important to hear arguments against the use of place-based systems in game AI, so that we can be sure they are the right tool for the job.

On the other hand, it may be that the apparent general blindness to place that many authors point out (which might arise, as Casey suggests, from place being such a fundamental part of the human world view that we do not normally think about it) causes us habitually to focus on the spatial at the expense of the placial. If this is the case then it would be right to give extra attention to place as a means of achieving our goals in game AI.

Furthermore, many of the sources consulted seem to be trying to rectify some perceived imbalance in the treatment of space/place/time, which perhaps implies that the area in general is very hard to balance properly, or is hard to define and analyze objectively.

## 2.3. Applications of Place-Based Study

There are various reasons for studying place, but they are all bound by a common theme of wanting to understand human interaction with space. Such understanding is of use to architects and planners

(who create places in the built environment), creators of virtual worlds (who seek to re-create the feeling of real places in virtual environments), and to geographers (who seek to present semantic data about space to human users).

In the field of Virtual Reality, the EC funded BENOGO project has researched how a sense of place might be re-created in virtual worlds (Turner & Turner, 2006) by creating simulated environments and analysing the subjective reports given by visitors to these environments in terms of four key properties of place (described below).

Another field involved in the creation of virtual environments is of course game AI, which has as its goal the appearance of realism, as opposed to the academic AI goal of actual realism. Following Damián Isla's mention of the lack of semantic information about space available to virtual agents in Halo2 (Isla, 2005), Adam Russell's article '*Turning Spaces into Places*' (Russell, 2008) investigated the area in detail, and suggested that place-based game world models could be a means of adding meaning and richness to the worlds of AI agents, leading to more believable behaviour (or, more specifically, helping to avoid unbelievable behaviours).

Some researchers in geographic information systems seek to use the concept of place to provide an enhanced user interface; to bring the spatial data in the system to the user in a more readily comprehensible form (Jordan et al., 1998).

Jones et al. have investigated how ontologies of place (an ontology in this context meaning a form of knowledge representation for specifying conceptualizations) might be used to service vague queries in geographical information systems (Jones et al., 2001).

Tanasescu & Domingue (2008) have also conducted research on applying concepts of place to local search, while Edwardes & Purves (2007) have investigated improvements to text-based image retrieval using placial semantics to define where a photograph was taken, by extracting data about place from images. In a similar vein, Hart and Dolbear (2007) from the Ordnance Survey describe the current cartographic/geographic research interest in the joining of geospatial data and semantic data in the 'Semantic Web'; web services based on semantic links.

Finally, fascinating experiments into augmented reality games on handheld computing devices (Martin et al., 2008) have mediated human subjects' very understanding of, and connection to, places – even to the extent of restructuring the available affordances. Affordance theory is discussed in section 2.6 below.

## 2.4. What is Place?

Edward Casey draws on the phenomenological tradition of Heidegger and Merleau-Ponty in his book *'Getting Back into Place'* (Casey, 1993). In it he opposes what he sees as an undue emphasis on the concepts of time and space, at the expense of place. He argues (convincingly) that place should instead be seen as "first among equals"; that without place, there could be no time or space, for place bounds everything, but nothing bounds place (except other places).

In Casey's view, 'implacement' is the fundamental way in which we experience the world, and the body is the vehicle through which we achieve it. This idea of 'lived place' is common in writings about place. The corollary to implacement is displacement, which gives rise to the idea that we need to be implaced to feel secure; once we have made a mental representation of place to overlay our raw sensory data about space we feel more comfortable, as if we have a map (although Casey does not use these terms). He highlights the key placial dichotomy of implacement and displacement, the difference between 'dwelling as residing' when implaced, and 'dwelling as wandering' when displaced.

Casey considers that experience starts from the body, rather than the mind, which is of particular relevance in determining directions and dimensions; concepts like 'near', 'far', 'here', 'there', etc. are all relative to the body experiencing them. This argument was also made by Henri Lefebvre (1974) when he rejected the fetishization (in the Marxist sense of overemphasis and oversimplification) of the internal mental representation of space/place, pointing out the contributions of both the physical world and the social forces of the prevailing culture.

It is worth noting that Casey's proposition that implacement occurs through the body is supported by work on body-scaled affordances. It has been shown that a connection exists between, for example, an observer's leg length and the perceived 'climbability' of a flight of stairs, even going so far as to experimentally determine a ratio between step height and leg length above which stairs are not perceived to be climbable (Jordan et al. (1998) citing Warren (1995)).

The geographer Tim Cresswell took a different approach to the study of place by attempting to analyze it in terms of transgressions against it (Cresswell, 1996). He describes places as a *discourse* between groups trying to define the meanings of spaces. He imagines such a discourse as a web or network, primarily created by the dominant cultural group, in which the meaning of a space and the appropriate behaviour in it are defined and linked together. A transgression is a act which challenges the meaning assigned to a place by the dominant group, and it is therefore characterized as 'dirty', 'mad', 'obscene', or simply 'out of place'. Ultimately, for Cresswell, place is about social power, which is a more political view of place than that found among philosophers like Casey. However, he

provides an excellent argument as to why some acts are considered to be 'out of place' and others are considered to be 'in place'.

Much of the geographic understanding of place and space seems to have its roots in 1970s sources and to be biased towards political or cultural study. Geographers tend to use the terms 'deterritorialization' and 'reterritorialization', coined by Deleuze and Guattari (Buchanan, 2005), which describe political/social displacement by dominant groups, reinforcing the emphasis on politics seen in Cresswell's work, as well as that of other cultural geographers. For constructing *models* of place, however, geographers turn to affordance theory (see section 2.6 below).

Jordan et al. (1998) provide a useful summary of place theory as it affects geographers, drawing on Entrikin, Curry, Relph and Cresswell. Curry (Jordan et al. (1998) citing Curry (1996)) sets out a number of concrete ways in which humans create places out of spaces:

- Naming: By giving a place a name we can discuss it and return to it.
- Categorizing: Identifying familiar elements (e.g. river, suburb) allows us to focus on what makes the place different from others.
- Creating or choosing a symbol: A symbol acts to condense the details of a place (e.g. pyramids = Egypt).
- Telling stories: Narratives give meaning to a place. This single point is the primary focus of Tim Cresswell's work, perhaps indicating that his focus may be too narrow.
- Doing things: Rituals help us to mark out our territory. Perhaps this is a sub-category of narrative; a way of creating narrative in a new place which has no existing narrative (for the newcomer).

An interesting psychological approach to place is that it can usefully be treated as an 'attitude' (Turner & Turner, 2006, citing Jorgensen & Stedman, 2002). This attitude would break down into a cognitive component (beliefs about the relationship between self and place), an affective component (feelings towards the place) and a conative/behavioural component (behaviours related exclusively to the place) .

Adam Russell (2008), drawing on Lefebvre, Merleau-Ponty and Goffman, also describes several key aspects of place:

- Places are neither entirely subjective, nor entirely objective. Russell suggests Lefebvre's representational space (which mediates between the social and physical realms) as a good model for mediating between an AI agent's local state and the world state.

- Places are integrated with behaviour, a concept expressed by the theory of affordances (see below).
- Places are hierarchical and concurrent. Unlike spaces, places are not discrete units but instead can layer and overlap each other. Different agents may not even be aware of all of the places existing in a space, and may experience entirely different places in a particular space.
- Places are associative structures. Russell cites Goffman's 'dramaturgical perspective', the idea that places are constructed according to the roles we are expected to play in them. However, it is worth noting that the idea that places can shape our actions is disputed by some in the urban planning field (Arefi & Triantafillou, 2005).

Finally, Turner & Turner (2006) conduct a review of place literature from phenomenological, sociological and psychological perspectives, identifying four key properties of place:

- The physical characteristics of the place (its space).
- The meanings and feelings associated with the place, including narrative about the place.
- The activities afforded by the place.
- The social activities afforded by the place.

Indeed, this seems to be an accurate summing up of thought in this area; most work on place could easily be related to one or more of these four categories. However, the following aspects are also encountered frequently:

- The dichotomy of 'in place'/'out of place'.
- The hierarchical, layered nature of places.

This project treats the above six points as being the key properties of place.

## 2.5. Postmodern Placelessness

One modern theme, inspired by Deleuze, is that of non-places; spaces that do not feel like places (e.g. fast food chain restaurants), often because of their uniformity. The term 'any place anywhere' is also used to describe this phenomenon. Ian Buchanan theorizes that this popular theme may be due to the 'hypermobility' of the postmodern subject having changed the way we experience place. This causes places that were not built in the postmodern era to seem 'placeless', when in fact it is simply that the observer is moving too fast to experience the place properly. A city like Las Vegas, on the other hand, only makes sense when viewed from a moving vehicle; since its history dictated that it

was experienced in that way (Buchanan, 2005). This theory serves to remind us again of the subjective (and 'situated', in the behavioural AI sense) nature of place.

It is now recognized in business and marketing circles that placeless places do not engage consumers. In particular, Gilmore & Pine II (2007) consider 'placemaking' to be the essential element for 'rendering authenticity' (which they claim is "the new business imperative") and cite the Jon Jerde Partnership (architects) as eminent practitioners in this field. 'Placemaking' in this context refers to the creation of rich experiences in spaces that distinguish them from other places. In Casey's parlance, the goal is to increase the implacement of the visitor.

## 2.6. Affordances

The concept of affordance was first proposed by James Gibson (1979), who suggested that we experience our environment through the possible actions it presents to us, not through its objective properties. Russell points out that this concept is closely related to von Uexkull's *umwelt* (the subjective world experienced by an organism), and is similar to notions of place, in that the affordances offered by an environment are neither wholly subjective nor wholly objective (Russell, 2008). He also points out that this model is already in use in games, citing the 'smart objects' of *The Sims* as an example.

Although affordances seem to be the means most often used to create models of place, Liqiu Meng points out that not all affordances are useful in all contexts (Meng, 2008). For example, a screen always affords touching, but if it is not a touch-sensitive screen then the touching will have no effect. To help create a sensible hierarchy and method for showing only appropriate affordances, she turns to the concept of reflex levels (citing Rosner and Schmauks, 2000). Reflex levels are tree-like structures containing meanings at different levels; a user seeing an image would first look for a meaning in the top reflex level, then might continue below that meaning to the meanings following from it (if they devote further cognitive processing time to the task).

Phil Turner has also commented on the difficulties of modeling affordances, and particularly the distinction between simple affordances, where the affordance *is* the context (citing Gibson), and complex (or 'perceived') affordances (citing Norman (1988)) where some extra thought or knowledge is required to *know how* to use the affordance (Turner, 2005). This is perhaps analogous to the meaning of an affordance being at a higher reflex level.

To account for these complex affordances, Martin Raubal (in the field of geographic information science) has presented an extended theory of affordances (Raubal, 2001). He suggests that affordances can be classified as either physical, social-institutional, or mental. Physical affordances

closely correspond to Gibson's original affordances; they are based on the physical properties of an object and a subject and may depend on body scaled ratios. Social-institutional affordances are derived from intangible social rules. For example, a road (physically) affords driving a car, but there may be a socially imposed speed limit as well as the physical limit of the car's performance. The third category, mental affordances, is directly derived from the other two categories. It refers to the agent's ability to choose among perceived physical and social-institutional affordances, based on the agent's own personality and cognitive processes.

Goffman's 'dramaturgical perspective' (Russell, 2008) seems also to be closely related to affordance theory, and could be viewed as a subset of the social-institutional affordances described by Raubal, as could the 'activity-oriented structure' used in *Black and White* (Evans & Barnet-Lamb, 2002). Evans and Barnet-Lamb proposed the use of 'activity' objects to co-ordinate group social behaviour in game AI agents. This representation has much in common with representations of place (describing a layered, hierarchical system of subjective relevance to agents) but does not explicitly include a spatial component. While defending their model against claims of behaviourism, they stress that the activities do not force agents to take part in them, they simply offer the possibility of activity and inform the agents of the consequences of engaging in it (or not). Further, "participation in an activity enables us to make choices we couldn't otherwise make". It can be argued that the activities described are examples of complex/social-institutional affordances; they afford the social activity itself and, once engaged in the activity, further affordances are made available.

Recently, Damián Isla has taken this idea even further in the AI Objectives System used in *Halo 3* (Isla, 2008). This system is designed to manage behaviour during complex battle encounters. It works by enumerating "tasks that need doing" and then assigning available agents to appropriate tasks. Isla notes that the tasks in the system are similar to affordances. The tasks are also structured using placial concepts like 'front' and 'back'.

## 2.7. Existing Models of Place

Jordan et al. present a model of place for use in geographic information systems, drawing heavily on the theory of affordances (Jordan et al., 1998). The reader will recall from section 2.6 that Raubal would later develop an extended theory of affordances in this context.

Their 'affordance-based model of place' consists of three elements: environment, tasks and agents. The environmental model is a hierarchy comprising several layers of increasing abstraction, which are used to manage and hide complexity (for example, the furniture of a building need not be considered when planning a route through a city). The task/action model is a hierarchy of

affordances, again comprising several layers of increasing abstraction. Finally, the agent model focuses on specifying the user’s capabilities. These three aspects combine to form a place definition, according to the agent, for a given task. Finally (as illustrated by Figure 1) they use a three tiered “Why, What, How” model to describe the combination of environment, task and agent.

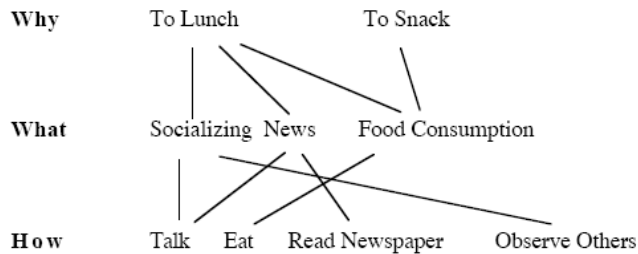


Figure 1: One possible Why, What, How level description of a Restaurant (from Jordan et al., 1998)

Jones et al. describe a model of place that combines spatial data (Euclidian distance) with a semantic hierarchy from which they compute a ‘distance’ between the topic of interest and the topic(s) linked to spatial locations (Jones et al., 2001). These two ‘distance’ measures are combined during a search to provide more relevant data to the searcher.

Tanasescu & Domingue (2008), similarly, have proposed a ‘differential’ model of place which is based on the same kind of semantic hierarchy, but uses the differences between two places as the key indicator of their identity. This approach resonates with Curry’s ‘categorization’ component of place (above).

In academic AI, Benjamin Kuipers’ ‘Spatial Semantic Hierarchy’ describes a knowledge model consisting of multiple, interacting, representations (Kuipers, 2000). The model is designed to mimic the human cognitive model, and is intended for use in the area of robotic map building. Each level of representation expresses only partial knowledge, which can be either qualitative or quantitative. The end result of the ‘Spatial Semantic Hierarchy’ is to reduce the environment to “a discrete set of distinctive states, linked by reliable actions” (Kuipers & Beeson, 2002). The agent can then create an internal map of these states and links, enabling it to navigate from one ‘place’ to another. Interestingly, a ‘place’ in this context is derived from the analysis/recognition of an image of a ‘distinctive state’, a set of such states being used to define a ‘place’ (see Figure 2).

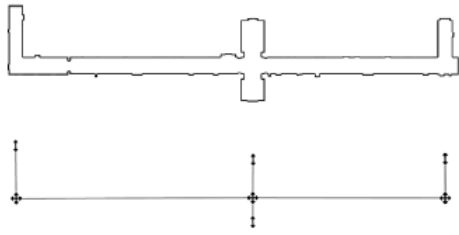


Figure 2: A physical environment and its causal/topological map. The map has 20 distinctive states and 7 places (image from Kuipers & Beeson, 2002)

In contrast to this reductionist approach, Strösslin et al. have presented a biologically inspired computational model of a rat's hippocampal 'place cells', which they have successfully demonstrated performing a path finding function on both real and simulated robots (Strösslin et al., 2005). Their model operates in continuous space using a neural network, a very different approach to that taken by Kuipers, which seeks to abstract continuous space into discrete states.

In game AI, a model of place was used in Halo 2 (Isla, 2005) that was described as a cognitive map overlaid onto a spatial navigation graph. It consists of a shallow hierarchy of spatial groupings; at the top level is an organizational component ('zone'), which contains 'areas', which are collections of discrete 'positions' derived from tactical analysis. The model contains no relational information and very little semantic information. Objects in the world are represented to the AI as affordance-like 'features' and physical 'volumes'. This model is much like the 'Spatial Semantic Hierarchy' described above, with the addition of affordance-like elements.

---

### 3. Research

The research in this project consists firstly of a proposal for a model of place for game AI, secondly a place agnostic behaviour specification (for agents in a Dwarven fortress) and, finally, two parallel implementations of the specification; a control implementation and an implementation based on the proposed model of place (“place-based implementation”).

#### 3.1. A Model of Place for Game AI

A model of place was designed as part of this project, specifically for use in game AI applications. Where a conflict exists between future extensibility of the model and provision of specific features, future extensibility has been favoured. The model therefore represents a general conceptual framework rather than any specific proposed implementation.

The model was designed in terms of the six key elements of place identified during the literature review (section 2.4 above); physical characteristics, activities, social activities, meanings/feelings, in/out of place dichotomy and hierarchy.

##### 3.1.1. An Overview of the Proposed Model

The model consists of a *place-world* which contains *places*, which have a *priority* and are containers for *affordances*, *limitations* and other *places*. The place-world is experienced through *implacers*, which are associated with a *body*.

- An *implacer* is owned by an agent and represents the agent’s subjective view onto the *place-world*, as well as allowing the agent to influence and change the *place-world*.
- The *body* associated with an *implacer* contains a set of *attributes* (such as the agent’s position, orientation, size, faction, age, etc.) that may affect how the agent perceives the *place-world* through its *implacer*.
- The *limitations* of a *place* affect whether an *implacer* can perceive the *place* (e.g. some places can only be perceived by a particular faction, or if the world is in a certain state).
- *Affordances* also have *limitations* that determine whether the *affordance* is available to a particular *implacer*.

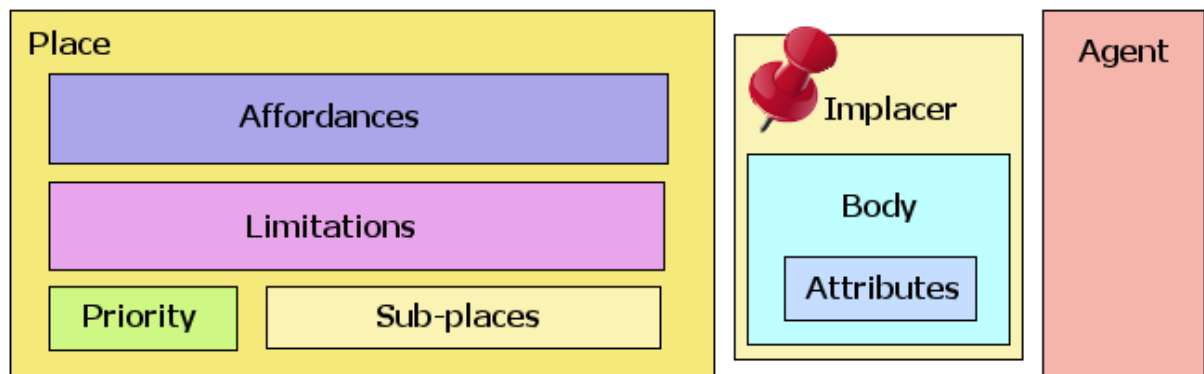


Figure 3: Overview of the proposed model of place

### 3.1.2. Agent, Implacer and Body

Because place is both a subjective and objective construct it is debatable where it should logically reside in a model; in the world or in the agent's mind. The proposed model treats places as 'objects' that exist in the objective reality of the world, but provide an 'implacer' object as a means of representing the agent's subjective view of them. The implacer is named after Edward Casey's notion of *implacement* (Casey, 1993). The principle of the implacer is analogous to a pin sticking into a notice board, and the places to pieces of paper on that board; the objects exist independently but share a point in common.

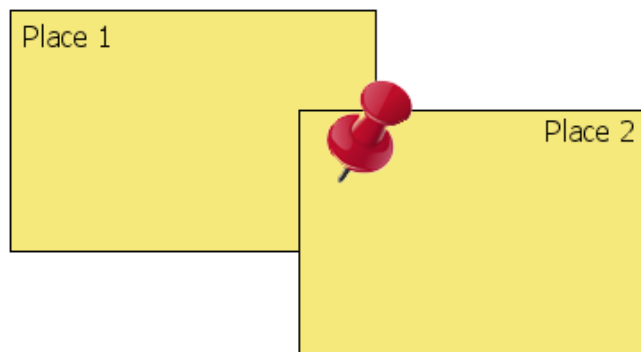


Figure 4: An analogy for an 'implacer', implaced in multiple places

The implacer is used to mediate the interactions between the agent and the place-world. Once created by an agent, an implacer is registered with the place-world. The agent is then responsible for configuring the implacer's 'body' correctly. The body is a repository in the implacer for information about the agent; 'attributes' such as size, faction etc. This information is then used by the implacer to filter out elements of the place-world that the agent cannot perceive, e.g. because it is of the wrong faction, or not tall enough. This also has the benefit of removing the agent from the model of place, leaving only the abstract representation of the agent (its implacer).

The implacer uses the ‘limitations’ associated with places and affordances to determine whether they can be perceived. Limitations may reference the implacer’s body (e.g. must be taller than 5ft), the world state (e.g. must be daytime) or other aspects of the place-world (e.g. must be no other implacers implaced, must be within a place owned by a faction).

It is not the function of the implacer to make any decision about which affordance (among those it presents to an agent) should be used; that function remains with the agent’s decision making routines.

Finally, the implacer may be implemented as a one way (read-only) view onto the place world, or as a two way (read-write) point of interaction with the place world, allowing an agent to dynamically change and reconfigure the place-world.

### 3.1.3. Space – the Physical Characteristics of the Place

Most literature on the subject considers physical space an important part of a model of place, although a notable exception is Evans and Barnet-Lamb (2002), which does not explicitly include a spatial component in its model of social activities.

The proposed model, while recognizing that a spatial component is important, does not explicitly define one. Rather, it provides a system of spatial constraints on each place that govern when it is available to be experienced. For example, if an implacer’s body:

- Is within range of a point.
- Is within a space (e.g. a collision shape, or bounding mesh).
- Is within particular bearings from a point (e.g. ‘in front of’).
- Has a particular orientation.

These constraints are expressed as *limitations* in the model. They therefore co-exist with other limitations, e.g. faction membership, world state, body capabilities, etc. In this way, the model allows for flexibility and even a certain ‘fuzziness’, rather than requiring a particular concrete spatial representation.

### 3.1.4. Place-world and Place Hierarchy

The place-world is the part of the model that represents a database of all possible places defined in the world, and their relationships to one another. It:

- Contains places.

- Allows agents to register their implacers (as consumers of place data).
- Provides implacers with access to the place data it contains.
- Is separate from the physical environment.

A place in the model acts as a container for affordances, limitations and other places. It therefore provides an implicit hierarchy (since it encompasses all places it contains). When places are found at the same level of this hierarchy, their relative importance is determined by a priority value.

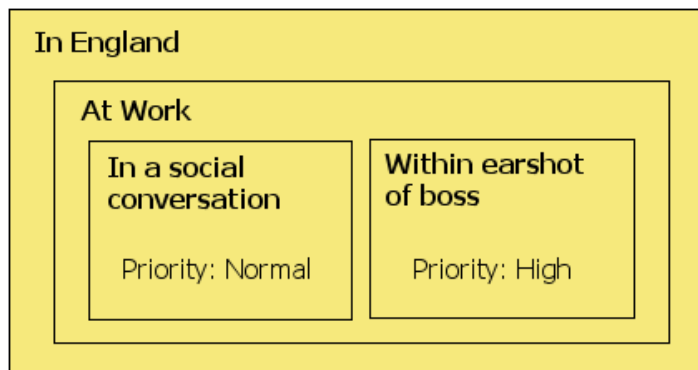


Figure 5: An example of hierarchy and layering

Places frequently overlap and interact with each other. For example, one may be simultaneously ‘in England’, ‘at work’, ‘in a social conversation’ and ‘within earshot of the boss’. The last two places may be incompatible; although the conversation affords gossip, the presence of the boss may prohibit not only gossip, but the conversation itself. The situation could be resolved by ending the conversation, whispering, or reconfiguring the conversation to an acceptable (non-social) topic.

Two approaches to this issue were considered; a hierarchy of places (where places contain other places and are composed in a tree structure), and a priority-based approach (where each place has a priority value).

The proposed approach is a hybrid of these two approaches, comprising a tree with 0 to n places at each leaf, each of which has a priority value to determine its importance relative to other places at the same leaf, as shown in Figure 5.

### 3.1.5. Affordances

As noted above, affordances are considered by many sources to be a fundamental part of the place-world, and are often used in the construction of models of place (especially in the field of geographic information science).

In this model, affordances are the key components of each place, representing anything the place offers, including (but not limited to):

- Actions available to agents
- Allowed actions
- Prohibited actions
- Social actions
- Reactions to actions
- Unavoidable effects on agents
- Knowledge or percepts available to agents

In Raubal's terminology (Raubal, 2001), the affordances in this model comprise both physical and social-institutional affordances. In Turner's terminology, they comprise both simple and complex affordances (Turner, 2005). By not explicitly distinguishing between categories of affordance, the model allows for maximum flexibility.

One category that is problematic is Raubal's mental affordances (which arise from physical and social-institutional affordances as part of the agent's cognitive process). Implementations of the model may choose to consider these as part of the place-world, as entirely outside the model (e.g. residing in the agent's decision making process) or some mixture of the two.

Meanings, feelings and narrative about a place are often cited as key properties of places. This model does not make specific provision for an explicit 'memory', but the behaviour of 'remembering' events can be approximated by adding affordances to places. For example, if one wanted to indicate that a member of a certain faction was killed in a place during gameplay (which would affect faction members' subsequent perceptions of that place) one could add an affordance to the place, available only to the faction in question, representing this knowledge.

### **3.1.6. Measures of Implacment**

The model allows for the possibility of calculating a metric for implacment, which could be used to affect the mood of an agent. A simple measure would be:

$$\text{Degree of Implacment} = n/N$$

Where:

- n = Number of affordances available (i.e. that the agent satisfies the constraints for)
- N = Total number of affordances in the places the agent is implaced in

For high degrees of placement the agent might be programmed to feel more comfortable (in place) and, for low degrees of placement, to feel anxious (out of place).

The model could also be extended to incorporate influence maps (Tozour, 2001) for each place as an additional spatial component. The influence map could then be used to provide a 'degree of spatial placement' metric to combine with the 'degree of placement' metric proposed above.

---

## **3.2. Detailed Methodology**

In accordance with the methodology described in section 1.2, a test case was designed that does not rely on any of the model's features or on any concepts of place. This place agnostic test case (described below) facilitates the parallel implementation and comparison of two behaviour libraries; a control implementation using no placial concepts and a place-based implementation using a subset of the features proposed in section 3.1.

The two implementations are provided to a client application through a common interface described in section 3.3. The control implementation is described in section 3.4 and the place-based implementation is described in section 3.5.

An extension to the test case was also designed (described below) and applied to each implementation.

### **3.2.1. Test Case/Behaviour Specification**

The test case is particularly focused on agent reactions and is set out as a behaviour specification for agents from two factions inhabiting a Dwarven fortress; workers and soldiers. The specification defines a number of spaces and the rules of behaviour associated with them.

Four types of space are defined within the fortress; work, military, social and dwelling. During the day, the actions permitted for each faction in each type of space change. Actions that are not permitted to an agent are considered to be prohibited, and will elicit a reaction from an agent observing them. The reaction depends on the factions of both the observer and the observed.

The full behaviour specification can be found in Appendix 4.

### **3.2.2. Extension to the Test Case**

In order to assess the extensibility of each implementation, and to investigate the ease of authoring and layering new functionality, the behaviour specification includes an extension; the user can

initiate a military inspection of the fortress. During an inspection (which may affect only part of the fortress), all soldier reactions are stricter and emphasise informal reprimands over reporting to superiors. The guarding schedule for soldiers is also extended to 24 hours a day.

The extension was applied to both implementations after the original behaviour specification had been implemented. The changes to the implementations that resulted from the extension were measured. The results are presented in section 4, but the specific steps required to author the extension for each implementation are detailed in sections 3.4.4 and 3.5.5, while the general steps required to author a new component for each implementation are set out in sections 3.4.5 and 3.5.6.

### **3.2.3. Evaluation**

The two implementations were evaluated in terms of their structure, complexity, performance, and memory usage. The results, and the metrics used, are presented in section 4.

---

## **3.3. Framework Implementation**

The software is divided into a client application and a behaviour library. The behaviour library itself is subdivided into a framework (described below) and two behaviour implementations.

In order to compare the two implementations fairly, a framework was implemented to hide them from the client application using Façade pattern (Gamma et al., 1995). The implementations and the client were not allowed to communicate other than through the framework.

The client application consists of a graphical user interface allowing the user to query the framework, change the active implementation and advance the simulated time, as shown in Figure 6.

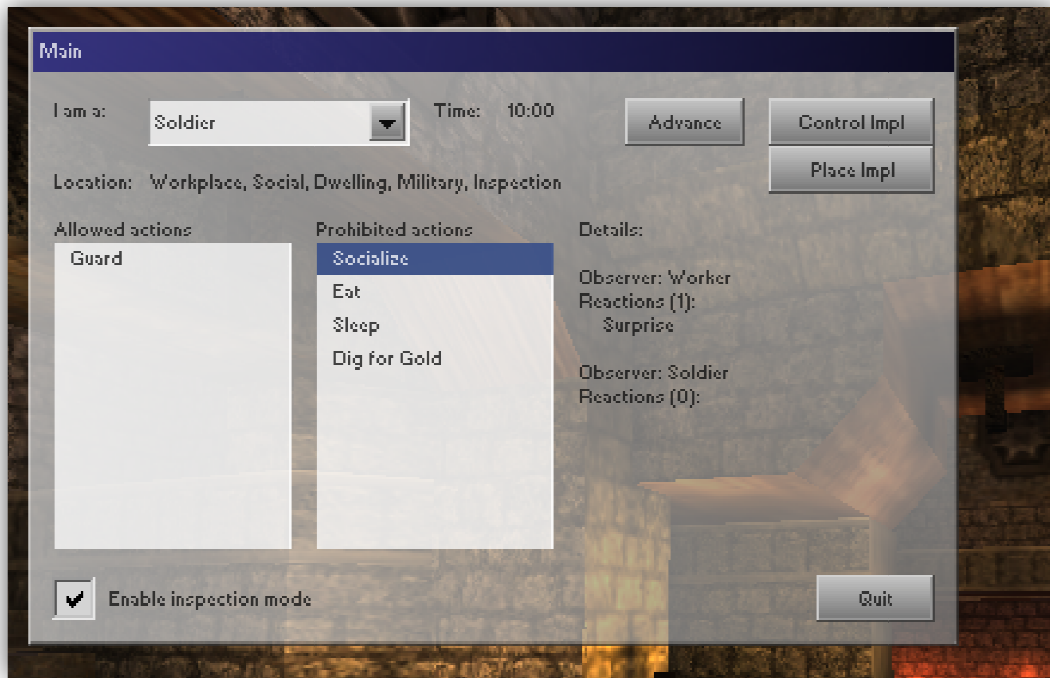


Figure 6: The client application's interface to the behaviour system

The Application Programming Interface to the behaviour system, used by the client application, is described by a set of enumerations (Figure 7) and the framework interface (Figure 8).

Behind the façade provided by the framework, each implementation implements a common interface (`Impl_Base`) as shown in Figure 9, making them interchangeable components that can be swapped at runtime. As far as the client is concerned, both implementations have exactly the same results and effect.

The key functionality required from each implementation is:

- Spatial representation
  - The ability to define a space with an axis aligned bounding box.
  - The ability to remove all spaces of a particular type.
  - The ability to find out what kind of spaces enclosed a particular world position.
- Action queries
  - The ability to find out, given the faction of an agent and its world position, what actions were allowed and what actions were forbidden to that agent.
- Reaction queries
  - The ability to find out an observer's reaction to seeing a forbidden ('out of place') action, given the faction of the observer, the faction of the actor and its world position.

The following subsections describe the framework in more detail.

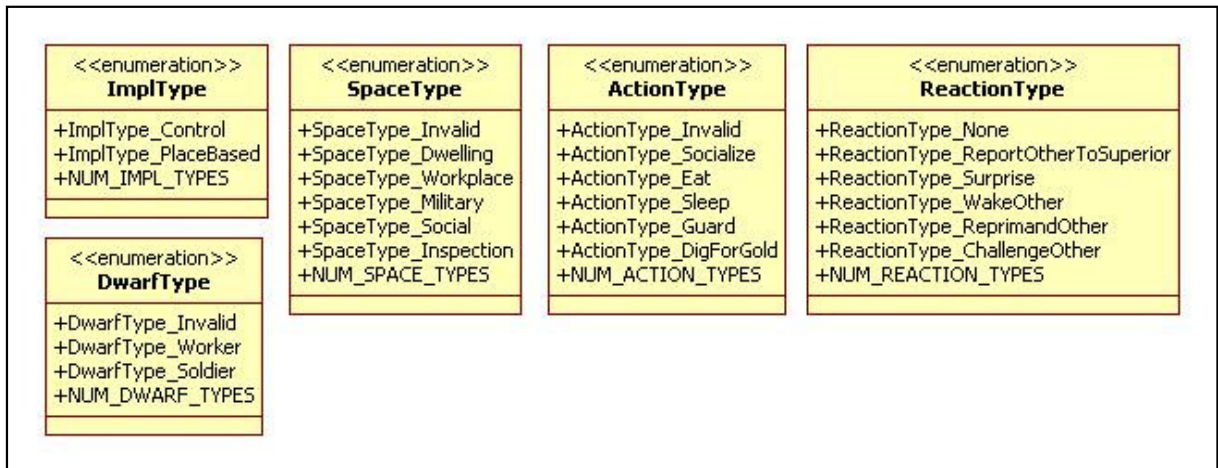


Figure 7: The enumerations used in the framework

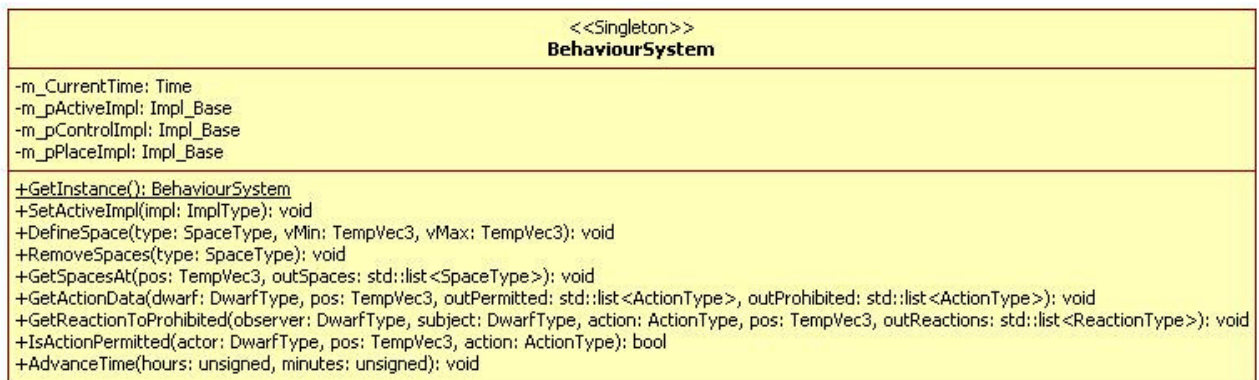


Figure 8: The framework/ façade’s API, as presented to the client application

### 3.3.1. Spatial Representation

The framework’s interface to the spatial representations of the implementations is defined by:

- An enumeration of the types of space available; `SpaceType`.
- `DefineSpace()`, a method that takes a `SpaceType` and two world positions defining the minimum and maximum extents of the space. Its effect is to define a space of the type and dimensions requested.
- `RemoveSpaces()`, a method that takes a `SpaceType` and causes all spaces of that type to be removed from the implementation.
- `GetSpacesAt()`, a method that takes a world position and a reference to a collection of `SpaceType`. The `SpaceType` of every space that encloses the specified position is added to the collection. Duplicates are allowed.

### 3.3.2. Action Queries

The framework that allows clients to query actions is defined by:

- An enumeration of the types of action available; `ActionType`.
- An enumeration of the types of agent faction available; `DwarfType`.
- `IsActionPermitted()`, a method that allows clients to determine whether a particular agent is allowed to perform a particular action at a given world position, at a given time. It takes a `DwarfType` representing the agent's faction, an `ActionType` representing the action of interest and a world position. It returns a `Boolean` value representing the result of the query.
- `GetActionData()`, a method similar to `IsActionPermitted()`. It takes similar arguments, but returns two collections of `ActionType`; one representing all permitted actions and one representing all prohibited actions. Each value of `ActionType` must be included in one, and only one, of the two collections.

### 3.3.3. Reaction Queries

The framework that allows clients to query reactions to seeing actions that are 'out of place' is defined by:

- An enumeration of the types of reaction available; `ReactionType`.
- `GetReactionToProhibited()`, a method that takes `DwarfType` values representing the factions of the observer/subject and the observed/object, an `ActionType` representing the type of action being observed, and the position of the observed action. It returns a collection of `ReactionType` values that the subject should display when witnessing the object performing the action 'out of place'.

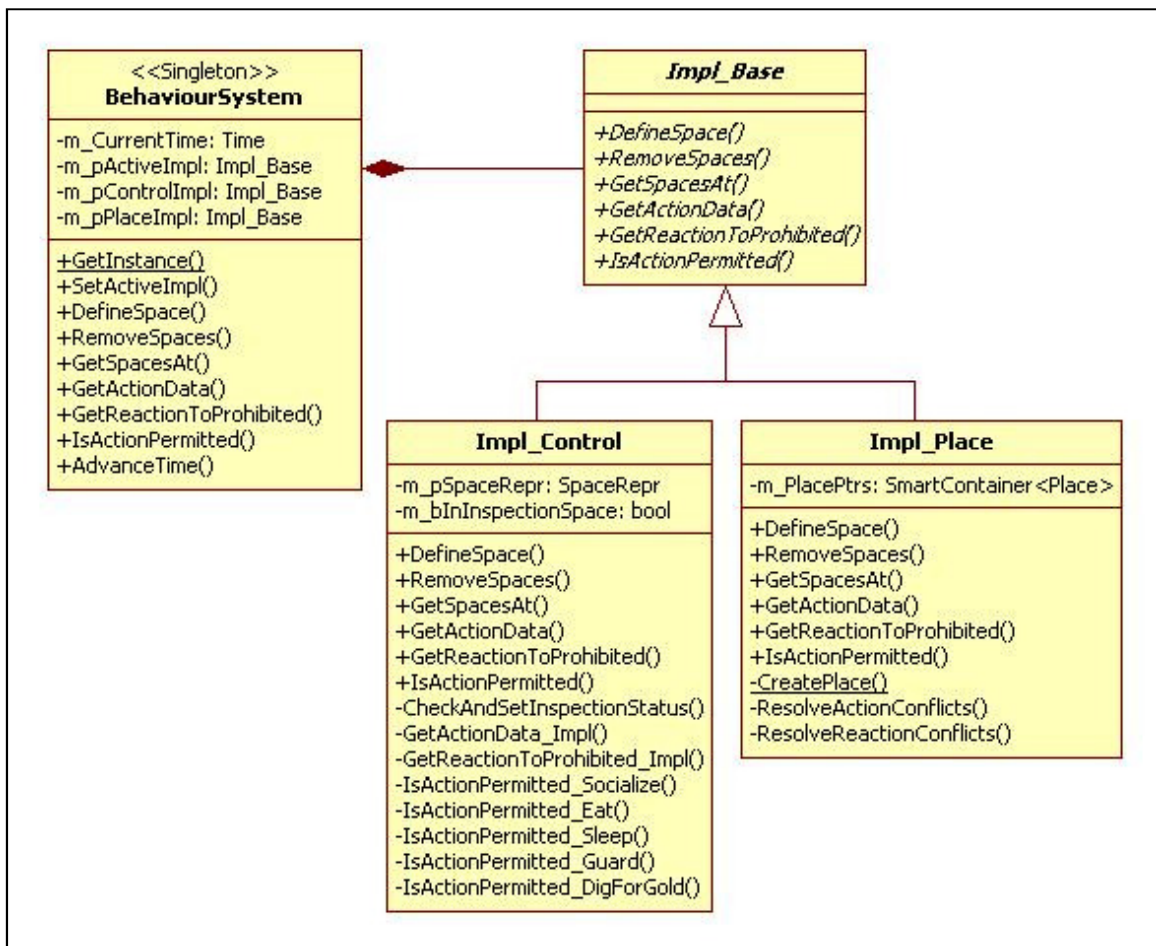


Figure 9: The design of the framework and interfaces to the implementations

### 3.4. Control Implementation

The guiding principle behind the control implementation was implement the minimum possible functionality required to satisfy the behaviour specification.

The following sub-sections describe how the required key functionality (section 3.3 above) was implemented, as well as how the extension specification was implemented and how new behaviours can be authored. Refer to Appendix 6 for a UML diagram of this implementation.

#### 3.4.1. Spatial Representation

A space is defined by an axis aligned bounding box (AABB). The control implementation maintains a container of AABBs associated with types of space. Queries on the spaces at a position are satisfied by iterating over the collection, testing the position against each AABB.

### 3.4.2. Action Queries

The `IsActionPermitted()` method delegates to one of five subroutines (one for each type of action) which encode the logic of the behaviour specification. Each subroutine checks whether the spatial requirements of the action are met, whether the action is available to the subject faction, and whether the action is available at the current time. If any of these checks fail, the subroutine returns `false` (the action is prohibited), otherwise it returns `true` (the action is permitted).

The `GetActionData()` method iterates over all defined actions. For each action, it queries `IsActionPermitted()`. If the action is permitted, it is added to the list of permitted actions, otherwise it is added to the list of prohibited actions.

### 3.4.3. Reaction Queries

The required logic for `GetReactionToProhibited()` is implemented using nested conditional tests; first on the subject, then on the object, then on the action. The implementation requires 7 `switch` statements and 26 `case` labels.

### 3.4.4. Extension Implementation

In order to implement the extension to the test case (a military inspection zone which alters the behaviour specification), action and reaction query methods must first call a private method (`CheckAndSetInspectionStatus()`) which takes a world position and sets a member variable (`m_bInInspectionSpace`): If the position falls within a space of type `SpaceType_Inspection` then the variable is set to `true`, otherwise it is set to `false`.

Conditional tests on the value of `m_bInInspectionSpace` are then used to determine whether the behaviour specification should be modified for a particular query. Five such tests were required to implement the extension to the test case.

### 3.4.5. Authoring

To author a new action one would have to implement another branch to the conditional logic in `IsActionPermitted()`, encoding the logic of the new action. One would also have to add logic to `GetReactionToProhibited()` to handle reactions to the new action, requiring four additional conditional branches.

To modify the behaviour in a space dynamically, one would have to implement conditional tests and alter the encoded logic based on the result, as was shown in section 3.4.4 above. As the number of

interactions increases, the complexity of this solution may be expected to increase exponentially, since the changes are not encapsulated.

---

## 3.5. Place-Based Implementation

The guiding principle behind this implementation was to incorporate a subset of the elements of the previously proposed model of place (section 3.1 above).

The following sub-sections describe how the required key functionality (section 3.3 above) was implemented, as well as how the extension specification was implemented and how new behaviours can be authored.

### 3.5.1. Overview

Refer to Appendix 7 for a UML diagram of this implementation. The key components derived from the model of place (described in section 3.1) that form part of this implementation are:

- `Place`
- `QueryState`
- `Affordance`
- `Limitation`

`Place` is an object containing `Affordance` instances (which represent actions and reactions), `Limitation` instances, an `AABB` representing the spatial component of the place, a `SpaceType` and a priority value. The core of this implementation comprises a collection of `Place` instances, configured to provide the behaviour required by the test case. For this implementation it was found to be sufficient to provide only one level of the hierarchy proposed in the model of place (section 3.1.4 above). This is implemented using the priority value associated with each `Place` instance. The ‘sub-places’ proposed in the model are not implemented.

In this implementation, the `DefineSpace()` framework method results in the creation of a new `Place`, configured appropriately for the behaviour of the desired `SpaceType`.

`QueryState` is a structure representing the state of a query in the system, including a subject and object `DwarfType`, the action being performed by the object, a world position and a time. As such it is a simple example of an *implacer’s body* component. This implementation does not explicitly include an *implacer* component because interaction with the client is mediated by the façade described in section 3.3, which to some extent stands in for an *implacer*. Queries made through the façade incorporate the state information that the model proposes should reside in a *body*. Therefore,

this implementation translates this state data into an instance of `QueryState` that the other components of the implementation can then use as if it were a *body*.

`Limitation` is a polymorphic interface providing a single method:

```
virtual bool DoesStatePass(const QueryState& state) const
```

Given an instance of `QueryState`, the method returns `true` if the state is sufficient to satisfy the limitation, or `false` otherwise. The implementation defines these concrete `Limitation`s:

- `Limitation_Subject`: satisfied if the `QueryState`'s object is of a particular type.
- `Limitation_Object`: satisfied if the `QueryState`'s subject is of a particular type.
- `Limitation_ObjectAction`: satisfied if the `QueryState`'s object action (action performed by the object) is of a particular type.
- `Limitation_TimeBetween`: satisfied if the `QueryState`'s time value falls between two particular time values.
- `Limitation_InvertDecorator`: constructed with an instance of `Limitation`, the result of which it inverts (i.e. `DoesStatePass()` returns `!pDecoratee->DoesStatePass()`).
- `Limitation_Or`: constructed with two `Limitation` instances, it returns the logical OR of their return values.

Concrete `Limitation` instances are used to control access to both `Place` instances and the `Affordance` instances contained in them.

`Affordance` is a polymorphic base class containing `Limitation` instances and a Boolean value indicating whether the affordance is forbidden. This value is used to allow `Affordances` in various `Places` to override each other while remaining loosely coupled. For example, the extension to the test case requires that certain reactions are no longer available during an inspection. This is implemented as a reaction affordance (marked as 'forbidden') in the `Place` representing the inspection, which overrides the reaction affordance already existing in the basic behaviour specification. The two concrete implementations of `Affordance` are:

- `Affordance_Action`: Represents the ability to perform a specific `ActionType`.
- `Affordance_Reaction`: Represents the ability to perform a specific `ReactionType`.

As proposed in the model of place, the `Limitations` in an `Affordance` determine whether an agent can perceive it; all limitations must be satisfied in order to perceive the affordance.

### 3.5.2. Spatial Representation

This implementation uses an axis aligned bounding box (AABB) to represent spaces. As such it does not follow the proposed model of place, which suggests using spatial *limitations* (contained in a *place*) instead of incorporating a concrete spatial representation in the model. This is an implementation detail; in this particular case it was known that an AABB was the only possible spatial representation allowed by the framework. The spatial representation could still have been implemented as an object derived from Limitation which contained an AABB instance, but it was considered needlessly complex.

### 3.5.3. Action Queries

Unlike in the control implementation, `IsActionPermitted()` is not the primary means of querying actions. Instead, `IsActionPermitted()` is implemented in terms of `GetActionData()`, which it calls and then determines whether the specified action appears in the list of allowed actions.

The algorithm employed by `GetActionData()` is as follows:

```
Configure a QueryState instance with the arguments passed;
Create temporary associative containers of Actions and Priorities;

for each Place:
  if (the Place's AABB encloses the test position):
    for each Affordance in the Place:
      if (the Affordance represents an Action AND the QueryState
          passes all the Affordance's Limitations):
        Make a pair from the Action and the Place's priority;
        if (the Affordance is forbidden):
          Add pair to the container of prohibited actions;
        else:
          Add pair to the container of permitted actions;

Resolve conflicts between Actions that appear in both the permitted
and prohibited lists, based on priority - if there is a prohibited
Action of equal or higher priority then the Action is prohibited.
```

### 3.5.4. Reaction Queries

Reaction queries using `GetReactionToProhibited` follow a similar algorithm to action queries (section 3.5.3 above), except that reaction affordances are considered instead of action affordances.

### 3.5.5. Extension Implementation

The behavioural logic in this implementation is not 'hardcoded', but instead arises from the arrangement of instances of the objects described above. As such, the extension required only an additional *place* definition; no changes were necessary to the existing definitions.

The additional place (`Place_Inspection`) has a higher than average priority value and contains:

- Four 'forbidding' reaction affordances (see section 3.1.5) that override affordances from the original behaviour specification inside this place.
- Two reaction affordances that add the functionality required by the extension.
- One action affordance (for `Action_Guard`) with a more relaxed time limitation than the one included in the original behaviour specification.

### 3.5.6. Authoring

No changes are required to the implementation to author a new action. Once the action is added to the framework's enumeration of available actions, it may be used to configure an `Affordance_Action`. The same is true for new reaction types.

To modify the behaviour in a place dynamically, one has the option of reconfiguring the existing places or providing a new place definition. As demonstrated by the extension to the test case, new place definitions can be layered on top of existing behaviour without requiring any changes to the existing behaviour.

---

## 4. Results

This section sets out the results of the research described in section 3, in particular the analysis of the control and place-based implementations described in sections 3.4 and 3.5.

### 4.1. Description of Objective Metrics

The key metrics used in this section are:

- Lines of code
  - Lines of framework code
  - Lines of test case specific code
  - Lines of extension specific code
- Cyclomatic Complexity
- Information Flow
- Memory usage
- Performance

When referring to ‘lines of code’, comments are excluded, as are all client application and façade code. Only the implementation specific code is counted. ‘Lines of code’ are further subdivided into ‘lines of framework code’, ‘lines of test case specific code’ and ‘lines of extension specific code’. The first category refers to code which does not specifically implement the details of the test case, but provides a framework for it to be implemented. The second category refers to code which implements or configures the specific details of the test case (not including the extension to the test case). The last category refers to code which implements the extension to the test case, including additional framework code. The total of the three categories must equal the total number of ‘lines of code’ described above (i.e. each line must be allocated to one of the three categories).

McCabe’s Cyclomatic Complexity is an indicator of a program’s control complexity, representing the number of independent paths through the program code (McCabe, 1976). While it offers a general indication of complexity and maintainability, it does not account for the flow of data through a program and can give misleading results in certain cases.

The Information Flow measure, by contrast, is derived from the number of interactions between modules in a program (Henry & Kafura, 1981). As such, it is better suited to analyzing data driven programs. Two divisions of this measure are calculated; visible and concrete. ‘Visible’ includes only those parts of the interface that are visible externally, while ‘concrete’ is limited to relationships that require recompilation of the client module if the supplier’s interface changes.

The program *C and C++ Code Counter* (Littlefair, 2006) was used to calculate the objective metrics described above. Subdivisions of the 'lines of code' metric were counted by hand (identified by specially formatted comments in the source code).

Memory usage was measured using the Microsoft Windows XP Task Manager and describes the average memory consumption (in KB) of each implementation. It was calculated as follows:

$$\text{Implementation specific consumption} = \text{total consumption} - \text{base consumption}$$

Where:

*total consumption* = the average (over 5 runs) consumed by the framework and the implementation together

*base consumption* = the average (over 5 runs) consumed by the framework alone

Performance was measured using automated, timed, unit tests of each framework method. The average execution time (in milliseconds) was calculated over 10,000 runs of each unit test. Because each unit test is different, there is no viable comparison between different methods in the same implementation; analysis is only meaningful for comparisons between the same method in different implementations.

## 4.2. Objective Metrics

Figure 10 sets out the objective metrics recorded.

Metric	Control	Place-based
<b>Lines of code</b>	826	1172
<b>Lines of framework code</b>	581	1078
<b>Lines of test case specific code</b>	225	80
<b>Lines of extension specific code</b>	20	14
<b>McCabe's Cyclomatic Complexity</b>	266	485
<b>Information Flow (visible)</b>	24	286
<b>Information Flow (concrete)</b>	1	476
<b>Memory Usage (KB)</b>	40	94.4
<b>Performance – Space Query (ms)</b>	0.014	0.013
<b>Performance – Action Query 1 (ms)</b>	0.087	0.17
<b>Performance – Action Query 2 (ms)</b>	0.202	0.065
<b>Performance – Reaction Query (ms)</b>	0.197	0.46

Figure 10: Objective metrics

### 4.3. Comparison

The following sub-sections compare the results for each implementation against each other.

#### 4.3.1. Structure

The control implementation comprises fewer lines of code in total, and has a lower proportion of framework code than the place-based implementation. However, the place-based implementation has a lower proportion (and quantity) of test case specific code and extension specific code.

It was also noted that the place-based implementation has fewer lines of comments than the control implementation, indicating that its structure is more self-explanatory.

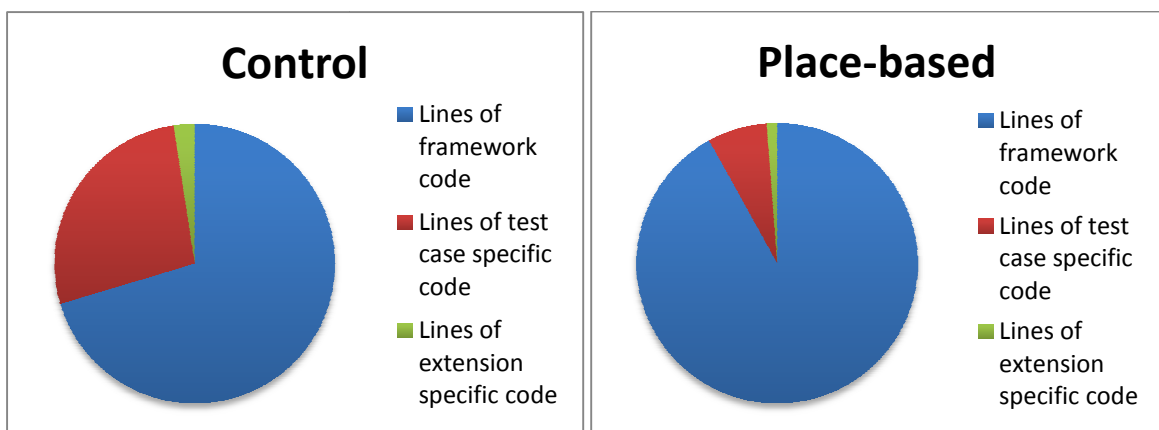


Figure 11: Code breakdown - proportional

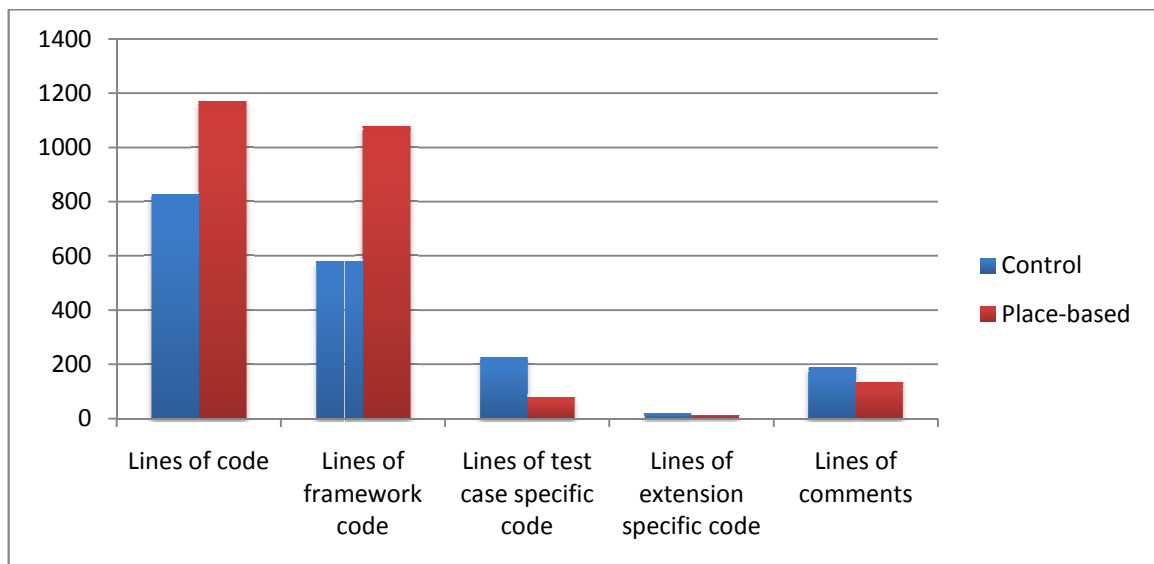


Figure 12: Code breakdown - quantitative

### 4.3.2. Complexity

Both the Cyclomatic Complexity and Information Flow measures indicate that the place-based implementation is more complex than the control implementation, as a whole.

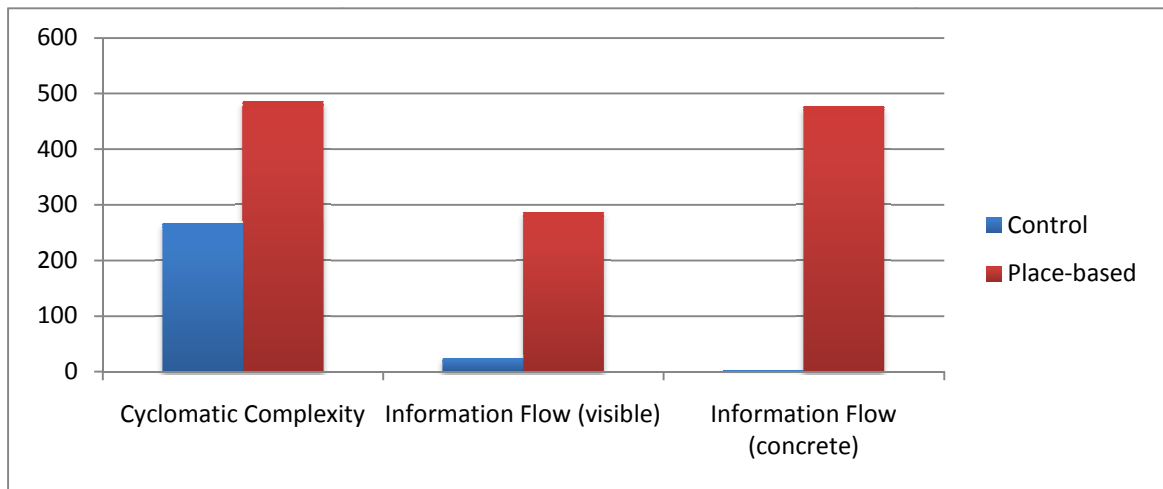


Figure 13: Comparison of complexity measures

However, as noted above, the place-based implementation has a higher proportion of framework code. Figure 14 shows the complexity measures apportioned to test case specific code only. While only an approximate measure of the complexity of the test case specific code, it shows a higher Cyclomatic Complexity for the control implementation together with more modest Information Flow values for the place-based implementation.

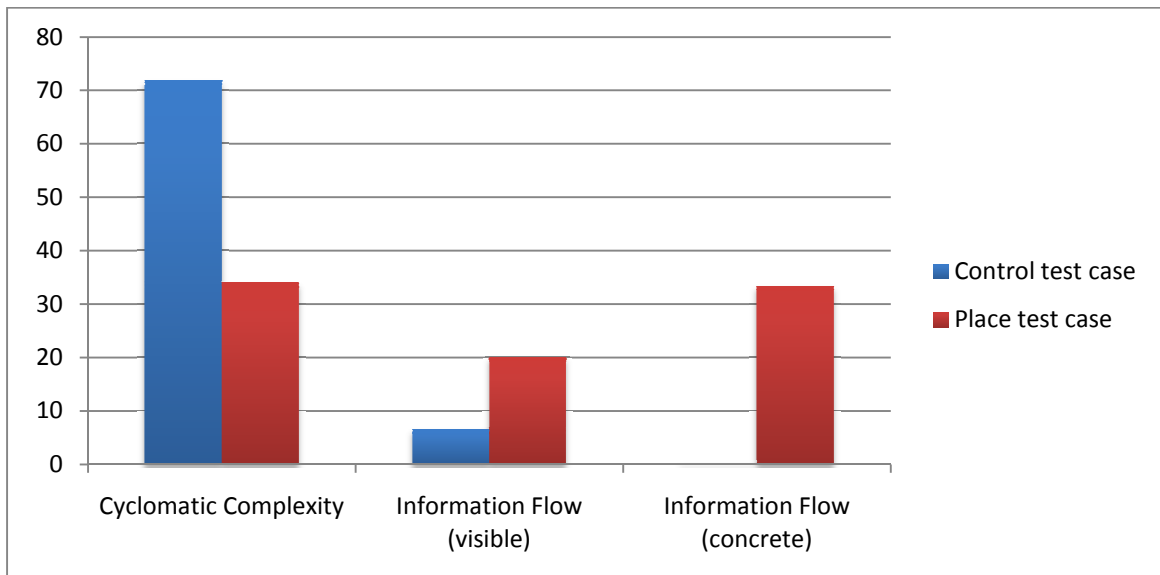


Figure 14: Complexity measures apportioned to test case specific code

### 4.3.3. Memory

The place-based implementation consumes more than twice the memory required by the control implementation. This is as expected, considering that the place-based implementation instantiates many more objects than the control implementation.

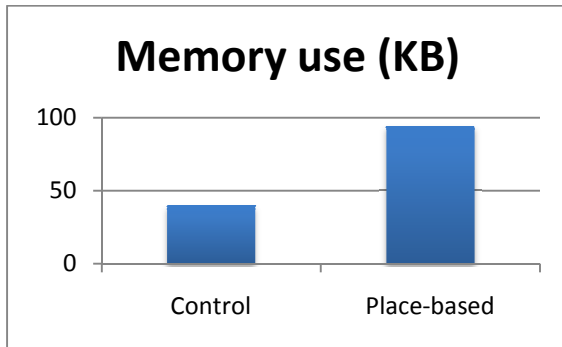


Figure 15: Memory usage – comparison

### 4.3.4. Performance

The control implementation performs best on the `IsActionPermitted()` and `GetReactionToProhibited()` unit tests, while the place-based implementation performs best on the `GetSpacesAt()` and `GetActionData()` unit tests.

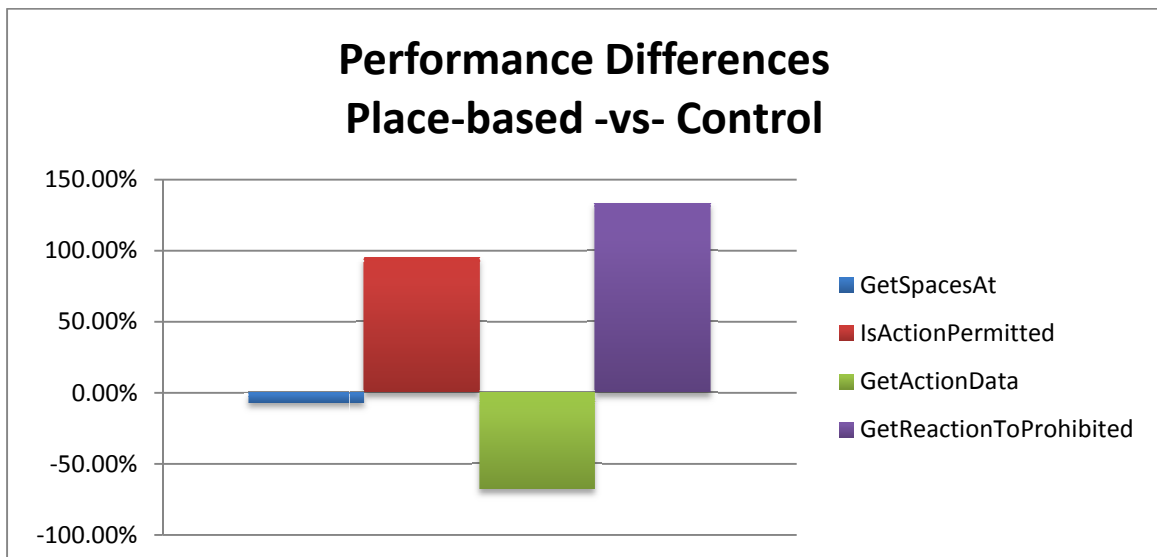


Figure 16: Performance differences between control and place-based implementations

These results indicate that the algorithm used in the place-based implementation is better suited to handling queries on the complete set of actions, prohibited and permitted, while the control implementation is better suited to handling queries on a single action.

Overall, the control implementation performs better than the place-based implementation.

---

#### 4.4. Projections Based on Extension Data

From the experimental data obtained it is possible to extrapolate data about how the codebase of a project might grow as further extension behaviours are added. Such extrapolation would be based on a number of assumptions and, since there was only one extension, the sample size is not significant. Figure 17 shows the projected total number of extra lines of code needed as the number of extensions increases.

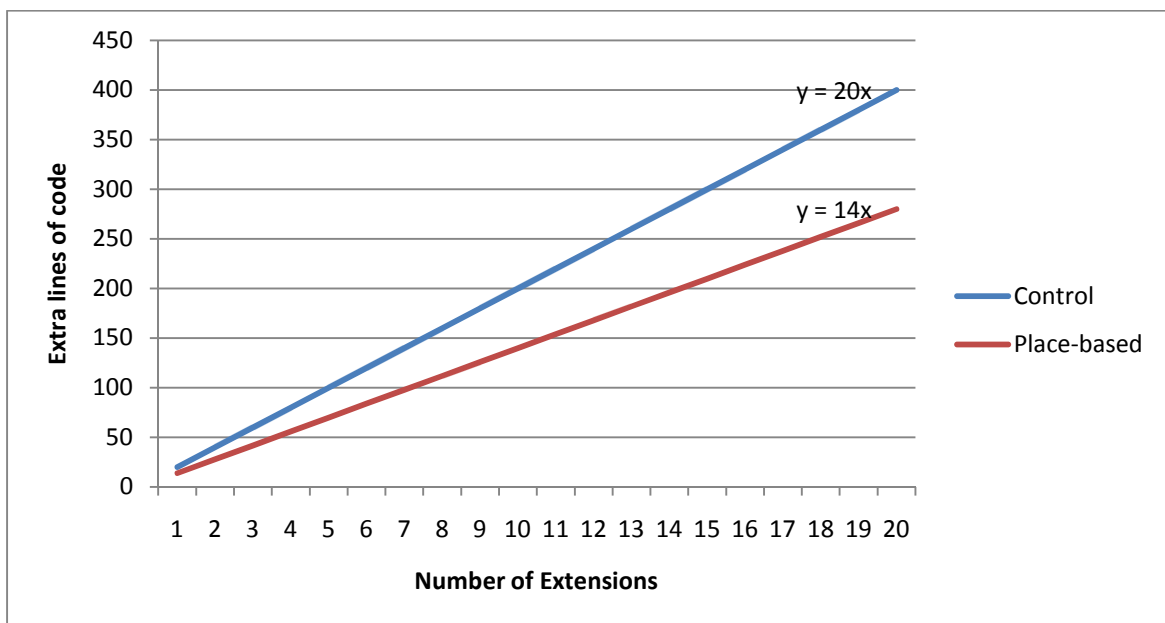


Figure 17: Projected code growth plotted against number of extensions

Although it grows more slowly, the place-based implementation starts with a larger codebase due to the extra framework code it requires. However, as the number of extensions grows there comes a point at which the size of the control codebase becomes larger than that of the place-based codebase. In Figure 18 it can be seen that this occurs after approximately 84 extensions.

It should be noted that the modularity of the place-based approach may be expected to offer benefits in terms of reduced complexity for such a large number of extensions. It is unlikely that the control implementation would be maintainable after 84 extensions. Further, the growth in codebase attributed to the control implementation may not be linear as the number of interactions between components rises.

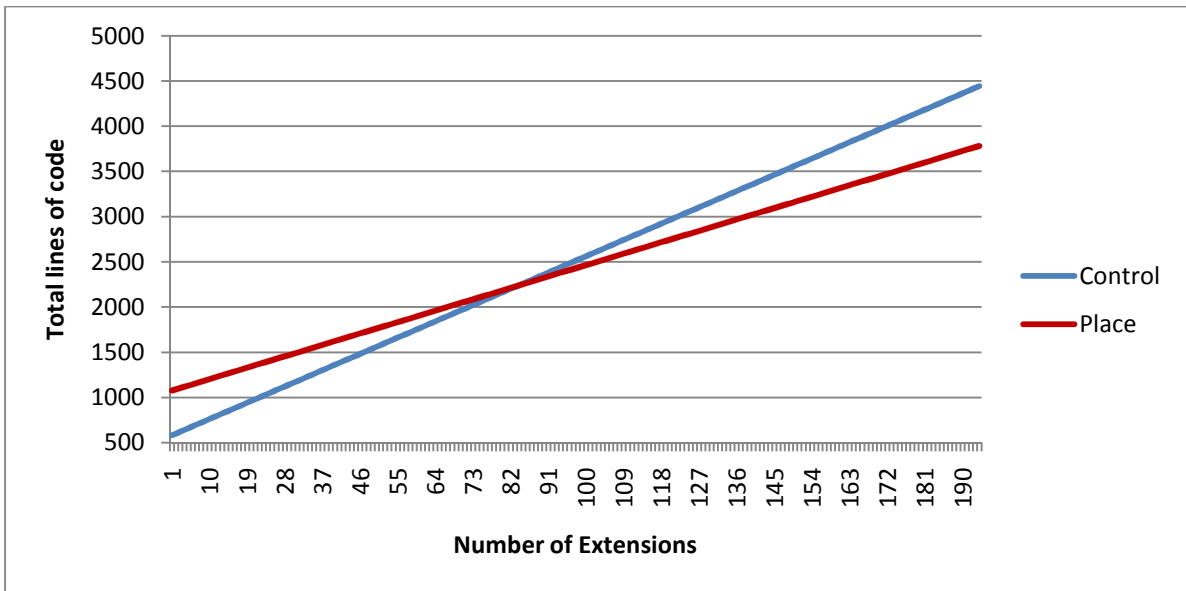


Figure 18: Projected total codebase size as number of extensions grow

## 5. Conclusions and Recommendations

As seen in section 2 (literature review) there is a trend in game AI toward supplementing traditional spatial world representations with semantic data about the world. Concepts of place have been proposed as a natural way to structure such semantic data. Further, affordances (section 3.1.5) have been a feature of most models of place proposed. It may be concluded that, whatever the eventual value of such models may be, concepts of place are influential in many fields of research, including game AI. The study of place and affordances may therefore be recommended as useful to game AI practitioners seeking to construct world representations for simulated agents.

In section 3.1 a model of place for use in game AI was synthesized, while the results of evaluating a subset of its features against a control implementation were described in section 4. It was demonstrated that the place-based implementation was more complex, performed worse, used more memory and required more framework code than the control. However, the place-based implementation required fewer lines of code for the implementation of both the test case itself, and its extension. Additionally, no changes to the existing codebase were required in order to implement the extension (unlike the control). Further, it is theorized that the complexity of the control implementation is higher when considered only in terms of the non-framework code. It may be concluded:

- That the place-based implementation is more extensible:
  - It is easier to author new behaviours.
  - It is easier to maintain behaviour implementations.
  - There is greater modularity of behaviour implementation.
  - Complexity and layering of behaviours is handled better.
  
- That the control implementation performs better:
  - It is generally faster (but see the comments in section 4.3.4 above).
  - It consumes less memory.
  - It requires less framework code.

The key recommendation of this report is that place-based implementations should be used judiciously. They have higher overheads but offer greater flexibility, ease of authoring and maintainability. On a constrained platform, or for simple behaviour specifications that are unlikely to change, the additional investment in infrastructure may not be justified. Figure 19 summarizes the considerations involved in deciding whether to use a place-based world model.

Use place model for...	Don't use place model for...
Relatively unconstrained platforms	Constrained platforms (performance, memory)
Medium to long projects (will see ROI on investment in place framework)	Short projects
Many behaviours	Few behaviours
Many interactions between behaviours	Few interactions between behaviours
Frequently changing behaviours	Relatively static behaviour specification
Easy authoring of behaviours	Infrequent authoring of behaviours

Figure 19: Guidance for suitable applications of place-based models

Finally it may be concluded that there are measurable benefits, in terms of authoring, to the use of a place-based world model. However, there are also a number of penalties. Therefore any decision must be based on the particular requirements of an individual project.

## 5.1. Criticism of Methodology and Future Directions

While it is believed that the methodology used was appropriate for the scope of this project, there are elements that could be improved upon.

The main point of concern is the integrity of the control implementation, and the validity of a comparison between it and the place-based implementation. It could be argued that, since the scope of the test case was so small, the comparison was not meaningful. Further, it could be argued that the comparison was, in reality, between a procedural implementation and an object orientated implementation of the same logic. Finally, it may be questioned whether the author remained impartial in the implementation of the control.

These concerns may be summarized as follows:

- The test case may have been too limited to be meaningful.
- The different programming paradigms used for each implementation may have affected the comparison.
- The author's bias may have contaminated the control implementation.

The methodology could be improved by taking an existing commercial game featuring complex agent interactions and extending it to make use of the proposed model of place. A more plausible comparison between the unmodified game and the extended version could then be made. However, the additional complexity of such a system might make a comparison more difficult.

Another concern is that the place-based implementation exercised only a subset of the features proposed in the model of place, making it arguable whether the model has been fully validated by this research. This was partly due to the place-agnostic test case, which was designed to be implementable without using any concepts of place. If the modified methodology proposed above were to be adopted, it would be possible to implement the model fully without compromising the control implementation (since it would already exist before the project began). This would also eliminate any suspicion of bias in the control implementation.

The contrast between the procedural programming style used in the control implementation and the object orientated style used in the place-based implementation is significant, since it is more likely that a commercial game would employ object orientated techniques, especially for designer authored behaviours. In this respect the control implementation has more in common with a behaviour system implemented in a designer authored scripting language. If adopting the modified methodology proposed above, it is suggested that the same programming paradigm is used for the extension as was used in the original game.

With regard to the test case, it could have been improved by incorporating more than one extension, and defining more reactions that are dependent on particular places. Also, a greater number of extensions would have allowed the projections in section 4.4 to be more accurate.

The constraints of this project did not allow subjective metrics to be investigated. However, it is theorized that any study of the subjective authoring experiences of developers would correlate to the objective findings of this report. Specifically:

- Lines of code may correlate to development time and ease of development
- Cyclomatic Complexity and Information Flow may correlate to extensibility and maintainability.

Another area that may be of future interest is the inclusion of concepts of place in AI library frameworks, or middleware, especially if an industry standard library emerges. Concepts of place are readily comprehensible to humans and may provide a way to better structure game AI authoring tools, as GIS researchers are already attempting (Jones et al., 2001).

---

## Bibliography

- Arefi, M., & Triantafillou, M. (2005). Reflections of the Pedagogy of Place in Planning and Urban Design. *Journal of Planning Education and Research* (25), 75-88.
- Axelrod, R. (2008). Navigation Graph Generation in Highly Dynamic Worlds. In S. Rabin, *AI Game Programming Wisdom 4* (pp. 125-141). USA: Charles River Media.
- Buchanan, I. (2005). Space in the Age of Non-Place. In I. Buchanan, & G. Lambert, *Deleuze and Space* (pp. 16-35). Edinburgh: Edinburgh University Press.
- Casey, E. S. (1993). *Getting Back into Place: Toward a Renewed Understanding of the Place-World*. Bloomington: Indiana University Press.
- Cresswell, T. (1996). *In Place/Out of Place: Geography, Ideology and Transgression*. Minneapolis: University of Minnesota Press.
- Dourish, P. (2006). Re-space-ing place: "place" and "space" ten years on. *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work (Banff, Alberta, Canada) CSCW '06* (pp. 299-308). New York: ACM.
- Edwardes, A. J., & Purves, R. S. (2007). Eliciting concepts of place for text-based image retrieval. *Proceedings of the 4th ACM Workshop on Geographical information Retrieval (Lisbon, Portugal) GIR '07* (pp. 15-18). New York: ACM.
- Evans, R., & Barnet-Lamb, T. (2002). Social Activities: Implementing Wittgenstein. *Proceedings of the Game Developers Conference 2002*. International Game Developers Association.
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- Gibson, J. J. (1979). *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin Company.
- Gilmore, J. H., & Pine II, B. J. (2007). *Authenticity: What Consumers Really Want*. USA: Harvard Business School Press.
- Hart, G., & Dolbear, C. (2007). What's So Special about Spatial? In A. Scharl, & K. Tochtermann, *The Geospatial Web* (pp. 39-44). London: Springer.
- Henry, S., & Kafura, K. (1981). Software Structure Metrics based on Information Flow. *IEEE Transactions on Software Engineering*, 7 (5), 510-518.
- Isla, D. (2008). Building a Better Battle: The Halo 3 AI Objectives System. *Game Developers Conference 2008*.
- Isla, D. (2005). Dude: Where's my Warthog? From Pathfinding to General Spatial Competence. *Artificial Intelligence and Interactive Digital Entertainment Conference 2005*.
- Jones, C. B., Alani, H., & Tudhope, D. (2001). Geographical Information Retrieval with Ontologies of Place. In D. R. Montello (Ed.), *Proceedings of the International Conference on Spatial Information Theory: Foundations of Geographic Information Science. 2205*, pp. 322-335. London: Springer-Verlag.

- Jordan, T., Raubal, M., Gartrell, B., & Egenhofer, M. J. (1998). An Affordance-Based Model of Place in GIS. *Eighth International Symposium on Spatial Data Handling*, (pp. 98-106). Vancouver, Canada.
- Kuipers, B. (2000). The Spatial Semantic Hierarchy. *Artificial Intelligence* (119), 191-233.
- Kuipers, B., & Beeson, P. (2002). Bootstrap Learning for Place Recognition. *The Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, (pp. 174-180). Alberta, Canada.
- Lefebvre, H. (1974). *The Production of Space (translated by D. Nicholson-Smith 1991)*. Oxford: Blackwell Publishers.
- Littlefair, T. (2006). *C and C++ Code Counter*. Retrieved April 10th, 2010, from Sourceforge: <http://sourceforge.net/projects/cccc>
- Martin, J., Mathews, J., Jan, M., & Holden, C. (2008). Restructuring activity and place: augmented reality games on handhelds. *Proceedings of the 8th International Conference Learning Sciences. 2*, pp. 35-42. Utrecht, The Netherlands: International Society of the Learning Sciences.
- Mateas, M. (1997). *An Oz-Centric Review of Interactive Drama and Believable Agents*. Technical Report CMU-CS-97-156, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA, USA.
- McCabe, T. J. (1976). A Complexity Measure. *IEEE Transactions on Software Engineering* , 2 (4), 308-320.
- Meng, L. (2008). To see and see through graphics - Toward affordance-driven geovisualization. *Proceedings of Virtual Geographic Environments - An International Conference on Development in Visualization and Virtual Environments in Geographic Information Science*. Chinese University of Hong Kong.
- Millington, I., & Funge, J. (2009). *Artificial Intelligence for Games, Second Edition*. Burlington, MA: Elsevier.
- Mori, M. (1970). The Uncanny Valley (Translated by Karl F. MacDorman and Takashi Minato). *Energy* , 7 (4), 33-35.
- Moriarty, G. (2000). The Place of Engineering and the Engineering of Place. *Techné: Research in Philosophy and Technology* , 5 (2).
- Norman, D. A. (1988). *The Psychology Of Everyday Things*. New York, USA: Basic Books.
- Olsson, P.-M. (2008). Practical Pathfinding in Dynamic Environments. In S. Rabin, *AI Game Programming Wisdom 4* (pp. 179-190). USA: Charles River Media.
- Paanakker, F. (2008). Risk-Averse Pathfinding Using Influence Maps. In S. Rabin, *AI Game Programming Wisdom 4* (pp. 173-178). USA: Charles River Media.
- Raubal, M. (2001). Ontology and epistemology for agent-based wayfinding simulation. *International Journal of Geographical Information Science* , 15 (7), 653-665.
- Raubal, M., & Moratz, R. (2008). A Functional Model for Affordance-Based Agents. In *Towards Affordance-Based Robot Control* (pp. 91-105). Berlin / Heidelberg: Springer-Verlag.

- Russell, A. (2008). Turning Spaces into Places. In S. Rabin, *AI Game Programming Wisdom 4* (pp. 71-81). USA: Charles River Media.
- Shaw, D., Barnes, N., & Blair, A. (2001). Creating Characters for Dynamic Stories in Interactive Games. *ADCOG 21: International Conference on Application and Development of Computer Games in the 21st Century*. City University of Hong Kong, HKSAR, China : <http://users.rsise.anu.edu.au/~davids/publications.html>.
- Straatman, R., Beij, A., & van der Sterren, W. (2006). Dynamic Tactical Position Evaluation. In S. Rabin, *Game AI Programming Wisdom 3*. USA: Charles River Media.
- Strösslin, T., Sheynikhovich, D., Chavarriaga, R., & Gerstner, W. (2005). Robust self-localisation and navigation based on hippocampal place cells. *Neural Networks* , 18 (9), 1125-1140.
- Tanasescu, V., & Domingue, J. (2008). A differential notion of place for local search. *Proceedings of the First international Workshop on Location and the Web (LOCWEB '08) Beijing, China*. 300, pp. 9-16. New York: ACM.
- Tanasescu, V., Gugliotta, A., Domingue, J., Villarías, L. G., Davies, R., Rowlatt, M., et al. (2006). Spatial Integration of Semantic Web Services: the e-Merges Approach. *International Semantic Web Conference (ISWC)*.
- Tozour, P. (2001). Influence Mapping. In M. Deloura, *Game Programming Gems 2*. USA: Charles River Media.
- Turner, P. (2005). Affordance as context. *Interacting With Computers* , 17 (6), 787-800.
- Turner, P., & Turner, S. (2006). Place, sense of place, and presence. *Presence: Teleoperators and Virtual Environments* , 15 (2), 204-217.

## Critical Evaluation

I am generally satisfied with the outcome of this project although, as mentioned in section 5, I would have preferred (if time had allowed) to employ a more thorough methodology. As it was, I felt that the scope of the project was too limited, given the scope of the topic. I am pleased that a model of place was synthesized, but disappointed that the scope of the project prevented me from exercising it fully. If I could do it again I would focus more on layering behaviours rather than agent reactions.

There are many sources of information on concepts of place, across a wide range of disciplines, and their study has been very interesting.

One point of difficulty was the high level of noise to signal found in the more philosophical writings, which might make this material inaccessible to game AI practitioners. In this respect I feel the industry could benefit from more literature reviews and summaries of thought in this area, and perhaps more multidisciplinary collaboration in the study of place. I am pleased with the literature review section of the project, which I feel provides a useful summary of thought in this area in fields which might not usually be considered by game AI practitioners.

While I found much material of interest in the field of geographic information science, the field of cultural geography (which frequently discusses 'place') yielded little useful material. Much of it seemed designed to be deliberately incomprehensible so as to disguise its lack of worthwhile content; the term 'place' is thrown around a lot by cultural geographers but rarely analysed in any useful way.

Geographic sources in general also seemed to me to place too much emphasis on the political meaning of place which, while valid, I feel is only one area in which we encounter places. The overemphasis on political values and motivations attached to places seems to me to pollute the discussion of place as a fundamental part of human experience. It almost seems like subverting a natural phenomenon to serve a political agenda, as well as being very narrow-minded. Perhaps it is for this reason that many of the geographic sources I read inspired quite a visceral feeling of distaste.

There is definitely a movement in the game AI field towards more complex, semantic representations of spaces (driven, I believe, by the increasing complexity of agent behaviour) and concepts of place have been proposed as the way to achieve this. From my experiences throughout this project, I tend to agree. Implication is the foundation of our experience of the world around us, and games have always striven to create the feeling of implication in virtual environments. As graphics technology improves, virtual worlds have become more and more realistic, yet virtual agents often appear to behave unnaturally; unbelievably. I suspect that the reason this is so jarring to a human observer is because it makes it apparent that the agent itself is not implied in its environment, a similar

phenomenon to the 'uncanny valley' in robotics (Mori, 1970). I believe that applying concepts of place to agent architectures, for the purpose of implacing agents in their environments, is the most natural approach to overcoming this problem.

Looking ahead, I believe that a better understanding of place will help to bridge the gap between humans and computers. At present, computers are often perceived to 'suck'. I believe a key reason for this is the fundamentally different ways in which humans and computers experience their environments and the difficulty of communicating effectively between them. While humans are implaced in the world through their bodies, computers have no such body – they are fundamentally abstract devices. Because of this, it is almost impossibly hard for a computer to understand the context of human experience, leading to difficulty in many areas (e.g. natural language processing, search, etc). The theory of place deals with human understanding of the world, therefore it follows that a better understanding of place will bring computer systems closer to their users and aid communication. This is already being investigated in the field of GIS.

Overall I have found this to be a very worthwhile project. I feel well placed to apply the knowledge gained during the project to my game programming practice, and expect to continue studying place in the future.

## **Appendix 1 – Project Proposal Form**

[This section is not included in the published version of this document]

## **Appendix 2 – Project Progress Sheets**

[This section is not included in the published version of this document]



## Appendix 4 – Behaviour Specification and Extension Notes

### *Setting*

The setting will be a small location with several rooms. Rooms will be designated as one of the following types of area:

1. Dwelling (e.g. house)
2. Work (e.g. mine, factory)
3. Military (e.g. turret, battlement, barracks)
4. Social (e.g. pub)

### *Actors*

NPCs fall into one of these categories:

1. Soldier
2. Worker

### *Actions*

Action: **Socialize/Relax**

Permitted at: Social

Permitted to: Worker, Soldier

Schedule: 17.00-22.00

Action: **Eat**

Permitted at: Dwelling, Social

Permitted to: Worker, Soldier

Schedule: 17.00-9.00

Action: **Sleep**

Permitted at: Dwelling

Permitted to: Worker, Soldier

Schedule: 22.00-08.00

Action: **Guard**

Permitted at: Military, Work

Permitted to: Soldier

Schedule: 9.00-12.00, 13.00-17.00

Action: **Dig for Gold**

Permitted at: Work

Permitted to: Worker

Schedule: 9.00-12.00, 13.00-17.00

### *Reactions on Observing Actions When Not Permitted*

Action: **Socialize/Relax**

Worker ->Worker: None

Soldier->Soldier: Report to superior

Soldier->Worker: Reprimand immediately  
Worker->Soldier: Surprise

Action: **Eat**

Worker ->Worker: None  
Soldier->Soldier: None  
Soldier->Worker: Reprimand immediately  
Worker->Soldier: Surprise

Action: **Sleep**

Worker ->Worker: Wake other  
Soldier->Soldier: Wake other, Report to superior  
Soldier->Worker: Wake other, Reprimand immediately  
Worker->Soldier: Report to superior

Action: **Guard**

Worker ->Worker: Surprise  
Soldier->Soldier: Surprise  
Soldier->Worker: Surprise, Reprimand immediately  
Worker->Soldier: None

Action: **Dig for Gold** (assume potential theft)

Worker ->Worker: Surprise, Challenge  
Soldier->Soldier: Surprise, Challenge, Report to superior  
Soldier->Worker: Surprise, Reprimand immediately, Report to superior  
Worker->Soldier: Surprise, Report to superior

### **Test Plan**

Player should be able to directly control a camera in a 3D environment. Player should be able to advance and reverse the simulated time of day at will. The following information should be displayed to the player:

1. The type of location at the camera's world position.
2. All actions permitted at the current world position for each actor role.
3. All actions NOT permitted at the current world position for each actor role.
4. For each such forbidden action, what an actor of each type would do if the action was observed at the world location.

Testing will be to determine that, in a specified location, the actions and reactions displayed are correctly reported, according to the above list of actions.

### **Justification**

The proposed test framework will allow a control implementation and a 'place model based' implementation that both meet the objective test criteria. The two implementations can then be compared and analyzed.

### **Extension**

To demonstrate ease of layering, treat the main implementation as, e.g., a behaviour system set up for the fortress and say we want to then add a new set of behaviours on top of that, e.g. for a quest

or special event. The new rules must co-exist with the existing ones, ideally without requiring a re-write.

Event: Military inspection

Effects: Within the inspection area, all soldier reactions are changed to be stricter and emphasise informal reprimands rather than reporting to superiors, and the guarding schedule for soldiers is extended to 24 hours a day.

Changes to reactions:

Remove 'report to superior' from these cases:

Soldier -> Soldier (Socialize)  
Soldier -> Soldier (Sleep)  
Worker -> Soldier (Sleep)  
Soldier -> Soldier (Dig for Gold)

Add 'reprimand' to these cases:

Soldier -> Soldier (Sleep)  
Worker -> Soldier (Sleep)

Changes to actions:

Soldier (Guard) has no time limits.

### *Extension notes - Place based*

As written it doesn't immediately support placing prohibitions on actions. It has no way to tell what to do if it encounters two affordances with the same result, but different limitations. At the moment it doesn't cross reference them, so if one of them passes then the action is available. This wouldn't work for something like redefining the reactions, which need to override what's there, rather than just add to it. It seems to me that we need some kind of priority hierarchy for the places themselves. Then if the same action is available to an agent from two places, it will only consider the highest priority one(s).

It seems there is a huge problem here with how to layer places on top of one another. Adding rules is no problem, but what about resolving conflicts?

1. We could choose not to have a separate place, but reconfigure the existing place.
2. We could have the superior place explicitly forbid actions in the lower places

In both cases the new place needs to know about the existing place, but I think the second option is marginally less invasive and more true to encapsulation and the principles of place. However, it does require changing the logic used to get reactions and actions to account for forbidden affordances.

Implementation was accomplished with the addition of two short methods to `Impl_PlaceBased`, addition of a priority enum to `Place`, adding a 'IsForbidden' field to `Affordance`, and creating a new place; `Place_Inspection` that consists of six reaction affordances (of which are 4 forbidden) and one action affordance. Not counting the system refactoring time, it took about 15 minutes.

Lines of code added (excluding framework): ~15

Ease of modification: Easy

Chance of logic bugs: Low

### *Extension notes – Control*

To make these changes I first added a layer of indirection to the `GetActionData()` and `GetReactionToProhibited()` methods. In the intermediate layer thus created, I checked whether the query position was within an Inspection space. If so I set a field on the instance, then call the original methods. I added special cases to the original methods to check this field.

I had to change a few different places, and I relied on comments to understand what was going on. As usual when modifying procedural code I had to understand how everything worked to change it. I could easily have missed something.

Lines of code added: ~25

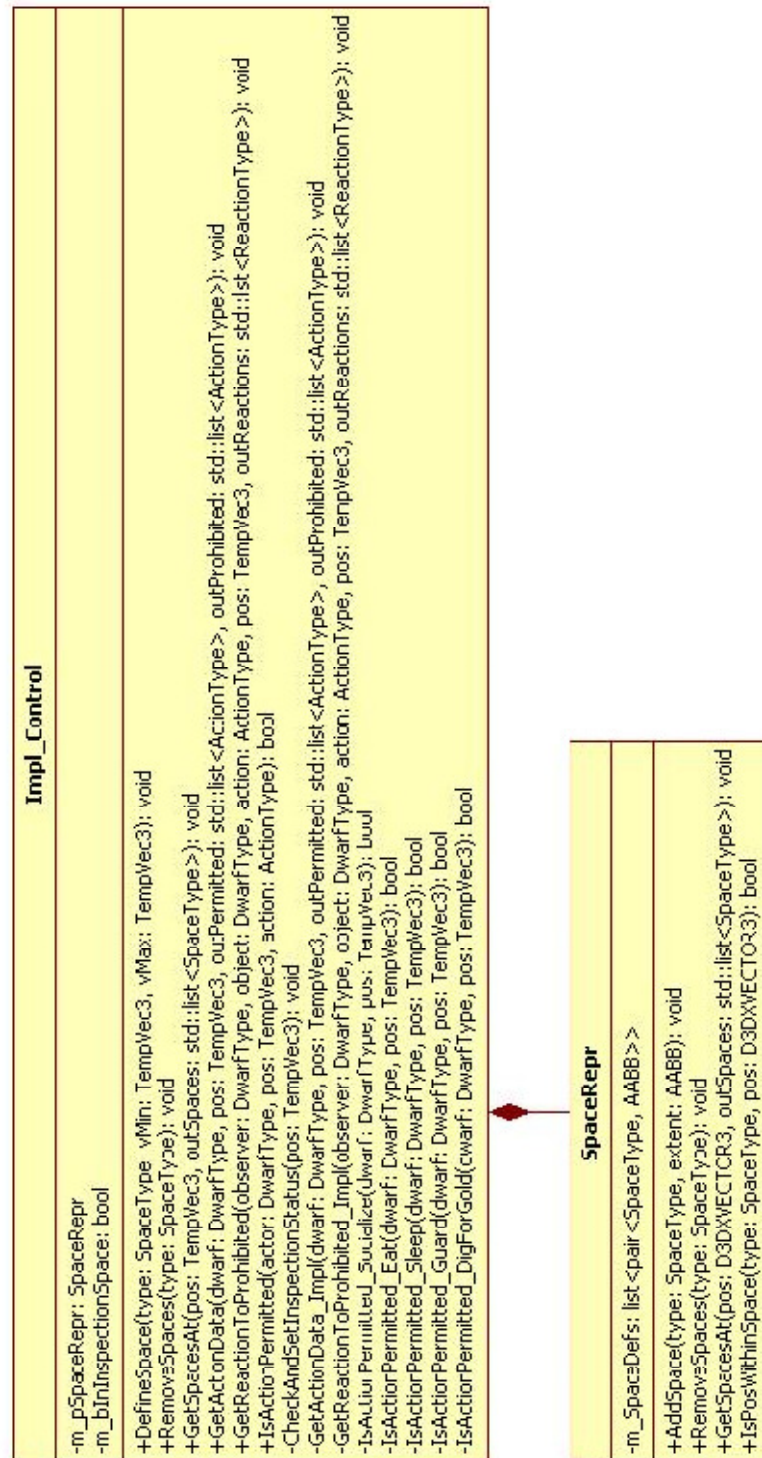
Ease of modification: Very Easy

Chance of logic bugs: High

## Appendix 5 – CCCC Analysis Data

	Control - overall	Control - per module	Place - Overall	Place - per module	Diff overall	Diff per module
Number of Modules	11		31		181.82%	
Lines of Code	826	75.091	1172	37.806	41.89%	-49.65%
Lines of Comments	190	17.273	133	4.29	-30.00%	-75.16%
McCabe's Cyclomatic Complexity	266	24.182	485	15.645	82.33%	-35.30%
Lines of code per comment	3.105		2.416		-22.19%	
Cyclomatic complexity per line of comment	0.714		0.274		-61.62%	
Information Flow measure	29	2.636	850	27.419	2831.03%	940.17%
Information Flow measure (visible)	24	2.182	286	9.226	1091.67%	322.82%
Information Flow measure (concrete)	1	0.091	476	15.355	47500.00%	16773.63%
Lines of code rejected by parser	102		160		56.86%	
Avg weighted methods per class	10.85714286		5.04		-53.58%	
Avg weighted methods per class (visible)	4.428571429		3.2		-27.74%	
Avg depth of inheritance tree	0.142857143		1.2		740.00%	
Avg No of children	0.142857143		0.68		376.00%	
Avg coupling between objects	3		3.24		8.00%	

## Appendix 6 – Control Implementation Diagram



# Appendix 7 – Place-Based Implementation Diagram

