Computing for construals in distributed participatory design – principles and tools

Meurig Beynon and Zhan En Chan

Computer Science, University of Warwick, Coventry CV4 7AL
wmb/echan@dcs.warwick.ac.uk

Abstract. Distributed participatory design aspires to standards of inclusivity and humanity in introducing technology to the workplace that are hard to attain. The demands it makes upon the development and use of computing technology are particularly topical, as the potential for automation and distribution through embedded and mobile devices continues to develop. Standard views of computation propose ways in which to interpret all products of computing as programs, but give limited conceptual support for understanding computer-based design artifacts whose role in communication and elaboration eludes capture in a functional specification. Empirical Modelling is a body of principles and tools that can be applied to the development of a variety of computer-based artifacts relating to analysis, design and use that are most appropriately interpreted as *construals* rather than programs. This paper, a revised and extended version of Beynon and Chan (2006), illustrates some of the ways in which Empirical Modelling principles can be used in developing construals that can assist distributed participatory design. The theme adopted for illustrative purposes is that of understanding the quintessentially British game of cricket, renowned for the complexity and subtlety of its rules and its rich concepts and vocabulary.

Key words: distributed participatory design, software development, Empirical Modelling, construals, cricket.

1 Introduction

Distributed participatory design (DPD) is concerned with design processes in which the stakeholders have different levels of expertise and competence and are located in different environments. The priority for this activity, as identified by Crabtree (2003), is the creative use of technology to improve working practices in such a way that it does not destroy the workers' skills, does not take away their autonomy, and enhances their quality of life.

Where the introduction of technology is concerned, DPD poses specific major technical challenges:

- **Integrating the human and the technological**: DPD demands an intimate integration of the technological infrastructure with human activities. As illustrated in Crabtree's account of the UTOPIA project (Bodker et al., 1987), the level of automation may not in fact be high; finding the most effective role for automation does not necessarily involve making radical changes to existing workpractices. Löwgren and Stolterman (2004:121-3) discuss the balance between the human and the technological with reference to control and autonomy, and observe that these are "not always handled intentionally by interaction designers".
- **Enabling flexibility and evolution**: DPD requires development techniques that make it possible to shape technology in a flexible and open-ended manner. Since participatory design engages with the entire development lifecycle (cf. Farshchian and Divitini, 1999), it should ideally be possible to take the perspectives of analysts, developers and users into account. This favours a conception of technology that is evolutionary in nature and accomodates elements of customisation and on-the-fly modification. Löwgren and Stolterman (2004:118-20) relate this

aspect of design to the development of pliable artifacts that "{support} exploration and fine-grained control" and are "flexible to unanticipated needs and desires".

- **Supporting diverse interaction and communication**: DPD highlights the role of technology in support of human interaction and communication rather than simply as a means to achieve goals in ways that are more efficient and cost-effective. The type of interaction and communication to be supported has also to be highly diverse. For instance, in addition to providing support for the workpractices to be implemented, it should also enable the role-shifting and exploratory investigation involved in the development (cf. Friis , 1988; Danielsson, 2004), where all stake-holders have an interest.

The primary focus of this paper is on the impact that two different underlying conceptions of computing technology have on distributed participatory design. In the sections which follow, our objective is

- to highlight ways in which a traditional conception of computing technology obstructs the integration of human and computer-based activities in support of DPD;
- to outline and illustrate an alternative conception of computing that offers principles and tools much better aligned to the emerging practice and aspirations in DPD.

Our illustrative examples will take the form of construals that have been developed using Empirical Modelling principles and tools in connection with understanding and simulating the game of cricket.

2 Conceptions of computing technology

The traditional way in which computing applications are conceptualized has deep historical roots. In using a computer as a calculator or data processor it is entirely appropriate to think in terms of categories such as: articulating the requirement and identifying it as a computable function; specifying this function in abstract high-level terms that can be understood by the programmer; translating this into code that can be interpreted by the machine; devising a protocol by which the user can supply parameters to the function and be presented with an output. The range of applications for computing has of course changed radically since such input-output processing was the dominating practical concern, but there has been no comparable conceptual shift in "understanding the fundamental nature of computing". This is not an issue that is of sole concern to the computer scientist; the pervasive role of computing, and the limited nature of our conceptual grasp, has arguably had a broad impact on our understanding of systems that combine human and automated agency.

The traditional conception of computer program is particularly ill-suited to meeting the technological challenges in DPD identified in the introduction:

- **Integrating the human and the technological?** In the classical view, an executing program is a means to an end in carrying out a computation – human activity and programs intersect at preconceived rendezvous points for stereotypical input-output interaction. At these points, the human has to act in highly constrained preconceived role as a 'user'. Activities such as "interrupting the program execution", or "modifying an executing program" are outside the conceptual scope.
- **Enabling flexibility and evolution?** In the classical view, the program requirement, its specification, its code and its user interface are different species, each with its own specialist human interpreter – the analyst, the designer, the programmer and the user. The fundamental idea behind the classical program is that once its function has been clarified and specified, every possible ingenuity and optimisation can be exercised in carrying out this function, thus saving computer system resources and user time. If you change the requirement, you may well

need to ditch the existing program, since ingenuity and optimisation is typically highly function-specific.

- **Supporting diverse interaction and communication?** The classical program is intended to support a black-box style interaction. It is not considered important for the user to understand what the program is doing internally, nor for the programmer to know what the user is doing beyond awaiting the computation of a specific function as mediated through a specific interface. Within the classical framework, it is possible to imitate cross-over interaction between different species of program specialist – for instance, to allow the user to adapt their interface, or customise the program execution for another user. But in so far as a computing application is viewed as a program, it has a fixed function that determines the way in which relationship between its execution and its surrounding context is to be interpreted, and this limits its expressed potential for communication.

The fact that the classical program – as we have characterised it above – is no more than a parody for what modern software technology discloses, is not the issue. Of course, advanced user interfaces and interface devices, databases, spreadsheets, agent technologies and the like have raised entirely new logistic, experiential and real-time issues that have transformed practice. The challenges we have identified all contribute to the goal of devising technology that supports *immersion* in the sense identified by Löwgren and Stolterman (2004:133-4). When Löwgren and Stolterman assert that "perhaps the most immersive activity in the digital realm is programming …", they can only be referring to those qualities of programming that emerge from its modern practice, and not to programming as it is narrowly conceived in computer science.

A re-conceptualisation of what is broadly termed "programming activity" is essential if we are to fully understand and exploit the emergent qualities of the best contemporary practice. It is also necessary because ways of thinking about computing technology are quite as influential as – if not more influential than – the technologies themselves. The influence of goal-orientation and programmed interaction remains prominent in research fields such as CSCW, user-centred design and agile development. It promotes a pernicious duality that separates the users of an application from its developers and partitions the life-cycle into distinct kinds of activity that are hard to integrate. This is in conflict with one of the central notion of DPD: that of treating workers as total human beings rather than merely users.

In dissolving the duality at the heart of classical thinking about computing, it is helpful to consider the distinction between two kinds of activity:

- identifying the potential users and functionalities for an application;
- gaining familiarity and understanding in the domain in which an application is to operate.

The concept of gaining familiarity with a domain is broader and more general than the identification of applications and users – it may not even presume any pre-existing notion of "user" or "application". It is also clear that the context for *merely* performing a task can be configured in such a way that its execution requires very little understanding of the domain. Indeed, as discussed at length in (Jullien, 1995), the importance of context in relation to task is a profound and long-established strand in Chinese thought that finds its expression in the concept of shi as an "inherent potentiality at work in configuration" (Jullien, 1995, p.14). The computer programming context can be regarded as an archetypal context that is contrived so that effective and meaningful action can be mediated by what Bornat describes as "completely meaningless" programs (Bornat, 2006).

Two quite different viewpoints on design activity are highlighted here. One puts its primary emphasis on clarifying the tasks to be carried out and bringing maximal ingenuity to bear on their efficient implementation through automation. The other focuses instead upon a less goal-oriented exploration of the potentialities within the domain of application. The first approach is associated with

the traditional conceptual framework that leads from requirements analysis of the workplace to specification, identification of function, implementation and maintenance. This framework involves an explicit framing of the notions of users and roles. The second approach involves a more open-ended holistic investigation of the application domain that cannot be decomposed into conceptually distinct formal phases. In this approach, the configuration of the application context, the nature of the problems to be addressed, and the roles for users and automation emerge in parallel.

3 Empirical Modelling

Empirical Modelling (EM) is a body of principles and tools that has been developed to support a conceptually different way of thinking about computing activity that is oriented towards a more holistic approach to design. Its primary emphasis is on studying dispositions within a situation and creating configurations to exploit these, rather than identifying and implementing functions to achieve pre-specified goals (cf. Beynon, Boyatt et al., 2006). To this end, it involves the development of computer-based artifacts that are more appropriately interpreted as *construals* than programs. That is to say, they are constructions made by the modeller that embody characteristic features of a situation that – like dispositions – are revealed through interaction.

In the DPD context, EM can be seen as targeting several of the objectives for "thoughtful interaction design" set out in Löwgren and Stolterman (2004). Like Löwgren and Stolterman, we view design as a process of mutual learning and knowledge construction, with particular emphasis given to primitive aspects of knowledge and learning that cannot be articulated in propositional form. Because of its emphasis on the incremental construction of digital artefacts that embody design knowledge, EM can also be viewed as an unconventional model of software development, and in this respect is well-aligned to Floyd's conception of software development as 'reality construction' (Floyd, 1992).

In supporting the design activity, the archetype for the construal developed using EM is an *interactive situation model* (ISM) that helps to capture and convey personal understanding of situations or phenomena. The referent for an ISM can be a real-world situation or phenomenon. It could also be an imaginary creation in the modeller's mind. The relationship between an ISM and its referent is mediated by the pattern of observables, dependencies and agencies that it embodies. The counterparts of observables and dependencies are variables and definitions within the ISM. Since the relationship between an ISM and its referent is appreciated primarily as an association given in the experience of the human observer (cf. Beynon 2005), many of the variables in the ISM have directly associated visualisations, for instance as geometric entities or display components and attributes. The current state of an ISM is represented by the current set of extant variables and definitions – a *definitive script*. This script may change and evolve dynamically subject to no constraint other than that the state of the ISM continues to stand in a meaningful relationship to its referent as far as the modeller is concerned. The counterpart of an atomic agent action, whether this is carried out by the modeller or any other agent, is a change to the ISM that involves adding, deleting or revising a definition in the script. A change to the definition of a single variable is effected in such a way that all contingent changes to the values of other variables are carried out atomically, so as to achieve an "instantaneous" update.

The significance of an ISM cannot typically be appreciated in isolation from our contextualized interaction with it; in this respect, it is unlike the classical computer program, whose semantics can be captured in the abstract functional relationships that it serves to establish. The interactions that shape an ISM have much in common with the kinds of experimental interaction associated with explanatory artifacts that a scientist might make in the early exploratory stages of an investigation (cf. Beynon and Russ, 2008), or that an engineer carries out when speculating about some aspect of a design. A characteristic feature of EM is that what we understand by a model is identified with patterns of interaction with such an artifact that emerge from experimental and exploratory activity over a period

of time. This activity is broad in scope: it may involve complementary interaction with the external situations and phenomena to which the ISM is intended to refer. It is also open-ended and potentially subjective in nature: the modeller's conception of an ISM and its associated referent is subject to evolve as the modelling activity progresses. These qualities of the ISM are well-matched to investigating what Löwgren and Stolterman (2004) identify as the *dynamic gestalt* of a proposed design. In comparison with traditional mathematical models, and in keeping with Löwgren and Stolterman's image of digital artifacts as pliable (Löwgren and Stolterman, 2004), an ISM is intrinsically fuzzy, soft and fluid. If its relationship to its environment can be sufficiently well-engineered, however, it may – at the discretion of the modeller – be exercised in a way that it emulates a model that is precise, hard and tightly specified (cf. Beynon et al. 2000, Beynon et al. 2006).

The principal tool for Empirical Modelling is the EDEN interpreter. The interpreter supports notations in which definitions to express dependencies amongst variables of various different types can be formulated. (For instance, variables may correspond to scalar quantities and attributes, strings, geometric elements like points and lines, components of a screen display, relational tables etc.) Distributed and web-enabled variants of EDEN have also been developed. These variants can be used to support modelling in a distributed environment.

The distributed variants of EDEN provide an open environment for developers and users to negotiate a shared interpretation and shared representation (cf. Beynon, Russ et al, 2006). They are well-suited to supporting Löwgren and Stolterman's vision of design as a process of debate (Löwgren and Stolterman, 2004:102) and mutual learning (Löwgren and Stolterman, 2004:152). Negotiation of meaning and consensus from many different personal, social and cultural perspectives is at the core of collaborative working. The meaning that is attached to an ISM is determined not by a formal computational semantics, but through the interactive experience and attendant associations it offers to each participant.

Distributed EM allows users and developers to interact through the modelling environment, whether through tweaking the ISM or redefining components entirely. This affords real-time collaboration among developers and users beyond document-centric communication. The EDEN interpreter offers the developer instant feedback, since any redefinition has an immediate impact on the state that is either directly visible to the developer (e.g. changes to the attributes of display components) or directly affects the disposition of the script to respond to future interaction (e.g. introducing a dependency to link the motion of a lever to the configuration of an object). As with traditional prototypes, this enables "potential users" to exploit an ISM in sense-making activities. But the modelling of behaviours as realised through first modelling dispositions (though typically consuming more attention and time and requiring more computational resources) is a much more expressive activity than directly prototyping behaviours. Because different design participants are free to exercise and interpret the dispositions embodied in an ISM in many independent ways, it is possible to give proper attention to what Löwgren and Stolterman (2004:167) identify as "the dilemma (and essence) of the design process: [moving] from diffuse and partially inconsistent visions to more specific and explicit operative images in order to debate understandings of problems and solutions". In particular, collaborative building of an ISM can respect Löwgren and Stolterman's "paradoxical" dictum to the effect that "there are no requirements that the design practice makes sense to users and designers in the same way, only that the designer sets the stage for a design practice so participation makes sense to all involved" (Löwgren and Stolterman, 2004:152) .

4 Illustrative ISMs for the game of cricket

Previous papers have illustrated ways in which Empirical Modelling has been applied to themes that relate directly to the agenda for DPD. They include: the collaborative development by D'Ornellas

(1998) and Sheth (1998) of a Virtual Electronic Laboratory as outlined in an earlier paper (Beynon and Chan, 2006) on which the present paper is based; the simulation of a historic railway accident using the distributed variant of EDEN (Beynon and Sun, 1999); and exercises in railway animation and lathe shaft design (Beynon et al, 1994). In this final section, our focus is on illustrating the role that constructing ISMs can play in supporting the characteristic features of design practice that are identified in the two quotations from Löwgren and Stolterman (2004) cited above. In particular, we shall sketch the way in which ISMs can be used to capture the "diffuse and partially inconsistent visions" that reflect sense-making from a variety of different perspectives, and indicate how these ISMs can in principle be refined and blended together to create "more specific and explicit operative images" and used to "debate understandings of problems and solutions".

For illustrative purposes, we shall review several ISMs that relate to the game of cricket, a notoriously complex game with a rich set of rules and associated jargon (see, for instance, the brief account of the game given by Woodbury (2002) in the John Hopkins University student newsletter). These ISMs have been created under many different auspices by a variety of modellers. They illustrate both collaborative and individual design activity that reflects a wide range of expertise in both cricket and in Empirical Modelling. Most importantly, they illustrate the wide range of different modes of observation that any complex activity typically admits, and the potential that EM affords for allowing such modes of observation to be captured and synthesised as ways of interacting with – and intepreting interaction with – a single artifact.

We preface our account of the ISMs with a short background section, in which we review the basic principles of the game of cricket and the general characteristics of the ISMs to be developed.

4.1 Background to the cricket ISMs

The basic elements of the game of cricket, as it might be played informally (for example, by children), are quite simple. The essential equipment consists of a ball, a wooden bat for hitting the ball, and a set of three sticks ("the stumps") that are placed in the ground in a straight line in such a way that the distance between a stump and its neighbour is just less than the diameter of the ball.

There are two teams in a game of cricket. In an official match, there are eleven players in each team, but in an informal game, there may be as few as four or five players in each team. The overall idea of the game is that the two teams take it in turns to score points ("runs") by hitting the ball with the bat, and the team that is able to score the most runs wins. At any given time, there are two players, one from each team, who stand in key roles. One is the player who currently wields the bat ("the batsman"), the other the player who throws (technically "bowls") the ball to the batsman. The batsman stands in front of the stumps, and the bowler bowls the ball from a point at right angles to the line along which the stumps are positioned, and aligned with the middle stump. In an official match, the distance between the bowler and the batsman is about twenty metres, but may be considerably less in an informal game. The players on the same team as the bowler ("the fielders") are disposed in a various positions around the batsman, anywhere from one metre to perhaps as much as seventy metres away so that they can retrieve the ball when it is hit.

For the batsman, the principal objective is to hit the ball with the bat in such a way as to prevent the ball from hitting the stumps and to ensure that the ball does not travel directly into the hands of a fielder without touching the ground ("a catch"). In order to score points ("make runs"), the batsman can run back and forth from the stumps to the point from which the ball is bowled as many times as possible before the ball is retrieved by a fielder and thrown back to the bowler or the stumps. There is a boundary around the playing area ("the pitch"); if the ball is hit in such a way as to cross this boundary – and whether or not the batsman runs – 4 or 6 runs are awarded to the batsman, according to whether or not the ball travels over the boundary without first hitting the ground. The batsman can continue to score runs in this way until such time as they fail to prevent the ball hitting the stumps

("are bowled"), or they hit a ball that is caught by a fielder ("are caught"). A batsman who is bowled or caught in this fashion is said to be "out", and has to leave the field to be replaced by the next batsman. The players in the batting team bat one after the other, and their scores contribute to a cumulative team total.

A little reflection on the basic rules described above reveals that there has to be some constraint on how many times the batsman can run back and forth. Indeed, the batsman is also deemed to be "out" if the ball is returned to the end point to which they are running before they arrive there (the batsman is "run out"). On this basis, a batsman ventures to run only when they suppose it plausible that they will complete their run before a fielder has time to return the ball to the appropriate end.

In building an ISM, the central idea is that of crafting a definitive script that captures the current state of a complex activity from the perspective of some observer. We associate the idea of 'understanding a complex activity' with being qualified to observe and interpret the activity from a variety of different standpoints. By applying this approach, it is possible to reflect the way that understanding typically operates at many levels and takes many forms. The merit of using EM principles to create several different ISMs, each associated with a different mode of observation, is that the scripts of these ISMs can be refined and blended within an open-ended interactive environment in a manner that is conceptually infeasible when seeking to combine traditional programs. Key steps in this blending and refinement include, for instance: combining the definitions associated with observing a particular state from several independent perspectives into a single script; adapting existing definitions and/or introducing new definitions to express the dependencies between these alternative views of state; identifying the different observers and modes of observation with specific kinds of agency and patterns of interaction with the composite script.

The ISMs for modelling observation of the game of cricket to be briefly discussed below exemplify the potential for development of this kind.

4.2 Understanding and observation

Cricket is as an archetypal example of a complex activity. We can view 'understanding cricket' as being equipped to observe the action in a cricket game in an informed way from some appropriate perspective. As in the case of any complex activity, there are many factors that determine the nature and quality of the understanding and of the associated observation. In respect of cricket, these include the expertise of the observer, their situation in relation to the game, and the degree of sophistication of the game itself. It will be useful at this point to briefly examine each of these factors in turn:

Expertise of the observer: The account of cricket given above is only sufficient to support a basic "school-child spectator" level of understanding of the game. Many observational details that are relevant to acting in the role of a batsman or bowler are omitted. Many features of the game that are critical in the professional game are only hazily represented. In a casual cricket game, a large batsman might stand in front of the stumps so as to prevent the ball hitting them without making any use of the bat, for instance. The laws of cricket allow a batsman to be deemed out in such a case ("leg before wicket"), but this law can be difficult to arbitrate in the absence of a disinterested observer who can judge whether such an obstruction has occurred. In all but the most informal kinds of cricket, there are impartial observers ("umpires") who are responsible for making such decisions. The ways in which an actual player, a spectator and an umpire attend to the game are quite different, and invoke different observables and protocols for response. To be an umpire, for instance, requires knowledge of special signs by which to communicate decisions to the scorers.

Situation in relation to the game: In watching a professional game of cricket, the location of the spectator can be quite critical. Your viewpoint determines for instance whether you are well-

placed to assess the accuracy or the speed of the bowling, for instance. The cricket ball is small, but the arena is large and the ball can travel at speeds in excess of 150 kilometres per hour. To be able to observe the ball is an acquired skill, and failure to observe is a common source of failures in understanding. A major cricket game can be played over several days, and is followed by people in many different contexts, most of whom are not present at the match. A game can be observed on television, followed via a radio commentary, or registered through a scorecard that records the play and is regular updated on a website. The nature of the observables in these contexts is quite different. On television, for instance the interaction between bowler and batsman can be observed in much greater intimacy than is possible when viewing from beyond the boundary, but the overall sense of the disposition and movement of the fielders is much harder to convey. To follow a cricket commentary requires detailed knowledge of a rich vocabulary relating to fielding positions, characteristic ways of hitting the ball, and bowling styles. Interpreting a scorecard involves learning appropriate conventions about how the actions of batsmen and bowlers are dynamically recorded in a suitable tabular form.

The degree of sophistication of the game: Many observables that are relevant to a professional game of cricket have no counterpart in a casual informal game. At any given time, one bowler is bowling the ball to a single batsman, but there is a second set of stumps at the point where the bowler releases the ball. There are also two batsmen – one located at each set of stumps – who bat in partnership and exchange places as and when they run, and two different bowlers who take turns to bowl in alternating directions from one set of stumps to the other. In the event of the ball being returned by a fielder to one of the sets of stumps before a batsman has completed a run, the relative position of the batsman becomes a significant observable in determining which batsman is deemed to be run out. In addition to two on-field umpires, who must arbitrate in respect of critical observations, there may also be a third umpire watching on camera who is able to advise with the help of technology. In some modes of the game, there are restrictions on how the fielders may be disposed about the field, and these restrictions take account of nuances such as whether a batsman holds the bat to the right or to the left of his legs (i.e. is a "right-handed" or "left-handed" batsman).

Taken together, the many modes of observation of a cricket match exemplify the kinds of 'diffuse and partially inconsistent visions', characteristic of design, to which Löwgren and Stolterman (2004:167) refer. And whilst it may seem that the state of the cricket match has a more clearly defined and objective status than the tentative conceptions of a design in the imaginations of the designers, observations of the match from different perspectives are not always easily reconciled. The terms that are used in giving a commentary on a cricket match, for instance, are not precisely matched to a specific physical location on the pitch or action on the part of a player. They already reflect an abstraction and interpretation by the commentator. Nor is the reality of the events that unfold entirely ambiguous. The role of the umpire is often to arbitrate about issues where there is room for genuine doubt, such as "Did the batsman hit the ball?", "Did the ball touch the ground before the fielder intercepted it?", "Did the ball reach the stumps before the batsman?". With the help of technology, it is sometimes possible to show the fallibility of umpire's judgement in a decisive manner. In exceptional cases, even the use of technology cannot answer a crucial question unequivocally, as when a critical event occurs between one video frame and the next.

In character, the resources that EM provides are well-suited to addressing Löwgren and Stolterman's agenda for design practice. An ISM supplies the stage for interaction and negotiation between human interpreters who are acting in different roles as agents and observers. It has a somewhat similar role and status to the technological artefacts that have been developed to assist the in-depth analysis of events in cricket. In deciding whether or not a batsman hit the ball, for instance, it

is possible to synchronise the movement of the ball as it passes the bat with an audio trace that is being simultaneously picked up by a microphone placed in one of the stumps, and to use an infra-red camera to detect hotspots where the ball makes contact with something in the vicinity of the bat. This potentially illustrates how designers can "debate understandings and solutions" through negotiation that is grounded in a synthesis of independent observations effected by digital artefacts.

5 Specific examples of ISMs for cricket

In this section, we give a brief account of several ISMs that have been developed using EM principles and tools. They respectively relate to: the very basic level of understanding of cricket that is expected of a spectator watching a casual game played by children in a park; to one aspect of the terminology that is required in understanding a cricket commentary; to the observation and understanding that is demanded of the scorer at a cricket match; and to the intimate knowledge of the physical characteristics of bowling and batting that is required to understand these skills more fully.

5.1 An ISM for basic cricket

The casual cricket watcher has a very basic understanding of the game such as was set out in section 4.1. To develop an ISM in which to embody this understanding, it is necessary to identify the nature of the observation that is required to appreciate the action in an informal game.

The first step towards this understanding is to appreciate what sequences of possible actions can take place between the point where the bowler prepares to bowl the ball, and the point where the ball is returned to the bowler to bowl the ball for the next time. Such a sequence of actions is somewhat confusingly termed "a ball", so that the entire play in a cricket match consists of a sequence of balls.

To characterise the pattern of possible actions that make up a ball in an ISM, we need to identify the significant observations that have to be made by the cricket-watcher. Informally, the bowler runs up the point where they release the ball towards the batsman, the ball travels to the batsman, the batsman then tries to hit the ball and the ball then travels to a point where it has either hit the stumps or been intercepted by a fielder. If the batsman hits the ball and the fielder intercepts it in the air before it hits the ground, this constitutes a catch.

This discussion highlights those observables to which the informed cricket-watcher pays most attention. They include, for instance, whether the ball has been hit by the bastman, whether it has touched the ground, whether it has been intercepted by a fielder, and whether it has crossed a boundary. In constructing an ISM to record the states that are encountered by the cricket-watcher in the course of a single ball, we can construct a script in which the variables represent conditions such as "the batsman has hit the ball", "the ball has hit the ground" etc. The following simple script fragment illustrates the nature of some of the relevant observables:

```
bowler_holds_ball = TRUE;
ball_live = FALSE;
ball_delivered = FALSE;
ball_received = FALSE;
ball_hit = FALSE;
ball_missed = FALSE;
ball_travels is (ball_hit || ball_missed) && !ball_intercepted;
ball_flies is ball_travels && !ball_rolls;
```

ball_rolls = FALSE;

ball_intercepted = FALSE;

ball_thrown = FALSE;

ball_caught is ball_intercepted && !ball_rolls;

ball_crosses_boundary is ball_travels && ball_dist >= 4;

The script describes the state prior to a ball being bowled. The variables in the script relate to observables such as whether the bowler has the ball, whether the batsman and fielders are in a state of readiness (as recorded by "ball_live"), whether the ball has been released by the bowler ("ball_delivered") and has reached the batsman ("ball_received") and been hit or missed by them. There is a significant distinction between the ball in motion on its way to the batsman and the ball in motion after the batsman has had received the ball and had the opportunity to hit it. There is an observable to record this distinction ("ball_travels") and this observable is defined by a dependency so that it first becomes true from the instant that the ball has been processed ("played") by the batsman. An additional observable records whether or not, after being played by the batsman, the ball is travelling through the air, or has already hit the ground ("ball_rolls"). Whether or not the ball is caught by a fielder is then dependent on whether or not it is intercepted whilst travelling through the air ("ball_caught").

A complementary set of observables is required to express the understanding of the basic cricket watcher. They relate to the ways in which the situation evolves from the state to state in the process of a ball being acted out. This evolution reflects state changes that are initiated by human agents, and those that are a consequence of natural laws. In an ISM, as discussed in section 3, all state-changing actions are mediated by redefinitions of observables. A ball hit by the batsman may spontaneously hit the ground as it falls under gravity, but it can only be caught through an intervention from a fielder. The informed cricket watcher is aware of the options for action that are available to the players within the rules of cricket. A fielder is not allowed to interfere with the ball until it has been played by the batsman, for instance. Specifically, the act of intercepting the ball is expressed by the redefinition:

ball_intercepted = TRUE;

and this action is only legitimate subject to ball_travels being true.

In building an ISM for basic cricket, we can capture the knowledge about the role of natural and human agency as governed by the rules of cricket by introducing additional observables. Conceptually, the framework of modelling the current situation by variables (representing observables) and dependencies (representing definitions), and registering agent actions as redefinitions is embellished by identifying agents and the latent actions they can perform and allowing the human modeller to play out a scenario by selecting which enabled actions are to be performed at any step. Note that in this embellishment, the term 'observable' is being used to refer to aspects of the situation of which the informed cricket watcher is directly aware, even though they may not be externally visible. For instance, they know that the ball cannot be intercepted by a fielder en route to the batsman, though this is not self-evident to a totally naïve observer.

Some simple extracts from the basic ISM will be used to illustrate these principles. We can consider the ball to be an agent in so far as its current status determines subsequent actions. In representing state-changing actions that can be performed by the ball, we can frame such definitions as:

```
action2 is ["is travelling to batsman", 2, ball_delivered && !ball_received];
action5 is ["is in the air", 5, ball_received && ball_travels && !ball_rolls];
action6 is ["is on the ground", 6, ball_received && ball_travels];
```

In these definitions, the right hand side is a list whose first component is an annotation that might be used in a commentary on the status of the ball, whose second is an index to identify the action, and whose third component is an enabling condition expressed in terms of other more primitive observables. For instance, the ball is travelling to the batsman when it has been delivered by the bowler but is yet to be received by the batsman. The repertoire of actions for an agent such as the ball is represented using a simple triggered procedure, thus:

```
proc ball : action {
        switch (action) {
           case 2: ball_received = TRUE; break;
           case 5: ball_rolls = FALSE; break;
           case 6: ball_rolls = TRUE; break;
           default: return;
       }
}
```

To play out a latent action, the modeller redefines the variable ‘action’. A list of enabled actions that are legitimately available for selection according to natural laws and the rules of cricket is defined and maintained by dependency. For instance, action5 above, which corresponds to the ball hitting the ground for the first time after having been received by the batsman, is only legitimate provided that the ball has not previously hit the ground – in which case ball_rolls is already true.

Though it may appear from the refinitions in the triggered procedure that the redefinition of ball_rolls when action 5 is executed is redundant, it serves a secondary purpose in updating an observable that records (in a very rough-and-ready way) the distance that the ball is deemed to have travelled since being played by the batsman. This update is effected by another triggered procedure:

```
proc inc_ball_dist : ball_rolls {
        if (last_ball_agent == "batsman") ball_dist++;
}
```

In modelling this updating action, other observables that distinguish the different phases in the playing out of a ball are used. These are expressed in terms of which human agent has discretion to change the state of the ball. The last_ball_agent is used to distinguish (for instance) whether the ball is en route from the bowler to the batsman, from the batsman to a fielder, or from a fielder back to the stumps.

As an illustration of how similar principles can be applied to the actions of human agents, consider the action of a fielder in intercepting the ball:

```
action7 is ["intercepts ball", 7, ball_travels && !ball_crosses_boundary];
proc fielder : action {
        switch (action) {
          case 7: ball_intercepted = TRUE;
                    break;
          default: return;
       }
}
```

In developing this basic model of the key observables, dependencies and agency that operate in cricket, the emphasis on giving priority to human-directed interaction and interpretation is evident. The aspiration is to make the relationships between variables, definitions and triggered actions in the ISM conform as closely as possible to what is experienced by way of observables, dependencies and agent actions in the real situation. An important aspect of this kind of model is that the modeller has discretion to take into account anomalous circumstances that arise in practice. For instance, it may be that a fielder declares that they have caught the ball even though in reality the ball first hit the ground. The effect of the umpire's decision might be to contradict this reality. Since the choice of which actions the modeller chooses to select at any point is merely discretionary, it is possible to explore and use the ISM to simulate such singular scenarios.

5.2 An ISM for cricket in a group context

The ISM for basic cricket sketched above was used as the basis for an undergraduate exercise on requirements analysis undertaken by a class of some 120 second-year students at the University of Warwick in 1993. The students worked in teams of 11 which were constituted in such a way as to include students with special expertise in leadership and management, programming, cricket and communication. They were asked to practise an EM approach to software development and yet follow a typical software project process which included regular meeting with the consultant (a senior student), internal team meetings, and quality assurance reviews. The objective of this exercise was to embellish the ISM for basic cricket from which the above fragments have been extracted so that it took account of richer scenarios and more subtle kinds of observation. Each student initially undertook this as an individual task, but the students came together to compile a team ISM from their personal ISMs. Teams used different strategies to carry out this exercise – for instance, some organised their individual modelling exercises in such a way that different subsets targetted the roles of the bowler, the batsman, and the fielder, or made a special study of issues such as modelling the process of running or the role of the specialist fielder who stands behind the batsman at the stumps ("the wicketkeeper").

In order to understand how different perspectives were integrated into the team's perspective and how the collaboration took place within a team during the project, we compared all the ISMs and analysed their reports, both individual and team, in a randomly chosen team ("Team A"). Our comparison showed that Team A was not significantly different from the other teams in terms of expertise and resources. Though they used different modelling strategies from other teams, Team A,

like the other teams, based their ISMs on the basic cricket ISM which was distributed to all teams at the beginning of the exercise. The analysis of Team A's ISMs followed the grounded theory approach (Strauss 1998). Initially, scripts of ISMs were partitioned and compared under three categories, namely, observables, dependencies for enabling action, and triggered actions. However, it became apparent from studying the scripts of their ISMs that the same scenario could be modelled by using a combination of observables, dependencies and triggered actions rather than exclusively using elements from one of the aforementioned categories. Consequently, fragments of the ISMs were tagged according to the scenario in a cricket game for which they give an account. Thirteen such scenarios were identified from the analysis of the ISMs, and they are listed in Table 1. The number of variants of each scenario generated by individual team members was counted and the level of detail was examined and given an appropriate rating. As might be expected, not every team member modelled all these scenarios in the same level of detail.

Scenario	Team	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11
a. How the ball is delivered	-	-	-	1	-	-	-	-	-	-	-	-
b. Abnormal deliveries	1	2*	/	2*	-	-	2	-	-	2*	-	1
c. How the ball is hit	-	-	1	-	-	-	-	-	-	-	-	-
d. How batsmen run after hitting or missing the ball	2	-	1	2	1	-	-	1	1	1	-	-
e. LBW and LBW appeals	2	-	1	1*	-	2	1	-	2	-	-	1
f. How batsmen are dismissed	-	-	-	1	-	-	-	-	-	-	-	-
g. How and by whom the wicket was broken	/	2	2	1	2	4	/	/	1	1	-	3
h. How the ball is intercepted	2	3	1	1	3	3	2	2	1	2	2	3
i. How the ball is caught	1d	3	1d	/	3d	3	2	2d	d	2p	/	3d
j. Modelling the transition between balls	1	3*	/	-	2	2*	1	-	-	/	2	2
k. Multiple batsmen and fielders	2	-	-	3	-	-	-	2	-	2	-	-
l. Scoring when the ball crosses boundary	1	-	1	-	1	1	1	2	/	1	/	-
m. Keeping scores	-	1	1	1	1	1	2	1	-	1	1	-

Table 1 – Contribution by the members in Team A

a. How the ball is delivered

One member of the development team, S3, modelled different types of delivery by the bowler. Although S3's account of the types of delivery was simple – merely represented by a few observables

and a few dependencies for enabling actions, exploring such scenarios required extensive observation of the situation. Unfortunately, S3's insight was not taken into account in the team's ISM.

b. Anomalous deliveries

There are two common anomalous types of delivery, namely, "no ball" and "wide ball". Although decriptions of both these can be found in any cricket manual (see e.g. The Laws of Cricket at Wikipedia), they were interpreted in significantly different ways by individual team members. The ways in which a wide ball was construed included: the ball itself veers wide (S2, S11); the umpire declares that the ball is wide (S1, S3, S9); the bowler delivers the ball wide (S6). The difference between these interpretations is the agent of the action, i.e. who is responsible for the anomalous action.

c. How the ball is hit

The individual ISMs reveal some misconceptions about the rules of cricket on the part of team members. S2 embellished the distinction made in the Basic Cricket ISM between the batsman hitting and missing the ball, deeming it significant that batsmen can hit the ball either when it is "in the air" or "on the ground". Note that this is not the same as distinguishing between a ball that is flying through the air or has already hit the ground after being hit by the batsman. There is a variety of ways in which the ball can be hit by the batsman (e.g. a "cut", or a "drive"). However, none of these was taken into account in the ISMs.

d. How batsmen run after hitting or missing the ball

Most of the individuals considered the possibility of multiple runs after a batsmen has hit the ball. The nature of the ISM is such that the state changes can always be played out by the modeller to confirm to an observed pattern. For instance, it is possible in this way to reflect the fact that a batsman who hits the ball is under no obligation to subsequently run. A more satisfactory model of the actual situation recognises that the decision about whether or not to run is one that is made by the batsmen themselves, and should be represented accordingly. S2, S3, S4, S7, S8, S9 model the fact that, having chosen to run, batsmen can choose to stop running while the ball is in the process of being fielded. Only S3 acknowledges that not choosing to run shoud be modelling as a possible action on the part of a batsman.

e. How batsmen are dismissed

There are in all 11 different ways in which a batsman can be declared "out". In most cases, the dismissal of a batsman requires a verbal appeal to the umpire by the fielding team. To reflect this, rather than making a model in which a batsman's dismissal automatically follows violation of a rule, S3 modelled the possibility that an appeal by the fielding side might be rejected. In S3's model, fielders *must* appeal in every possible situation where a batsman may have been dismissed in order to get the official endorsement of the umpire. All other team members formulated a direct link between the dismissal of a batsman and four common scenarios for dismissal.

f. LBW and LBW appeals

The idea of the Leg Before Wicket (LBW) rule is to prevent the batsman from using his body to obstruct the ball so that it cannot go on to hit the stumps. Judging whether a batsman should be given out LBW is not straightforward in most cases. This was not reflected in many of the individual ISMs. Most individuals in the team (S2, S3, S5, S6, S8, S11) deemed that if the ball hits the batsman's leg, this will always result in the dismissal of a batsman. Only S5 and S8 have observed that an LBW decision can only be given in response to an appeal from the fielding team.

g. How and by whom the wicket was broken

Hitting of the stumps by the ball ("breaking the wicket") plays a critical role in many aspects of cricket. Observing the fact the wicket has been broken is a straightforward matter, but more than a casual observation is needed decide which player agent should be credited with breaking the wicket and how this came about. Most of the individuals in the team observed that the wicket is broken by the ball when a batsman is "out" bowled, but only five of the team (S2, S4, S5, S9, S11) considered the possibility that the wicket can be broken after the ball is fielded (as when a batsman is run out, for instance). For instance, S5 and S11 observed that the bowler or the wicketkeeper can break the wicket by throwing the ball at the stumps. Others modelled the legitimate possibility that the wicket can be broken by the wicketkeeper's hand provided that it currently holds the ball (S4, S5).

h. How the ball is intercepted

The fielders and the wicketkeeper are most commonly the agents who intercept the ball. Most of the individuals in the team observed this in their individual ISMs. A few individuals (S1, S4, S5, S7, S11) also observed that the ball can be intercepted by the bowler. However, none of them acknowledged that the ball can also be intercepted by the umpire (albeit accidentally).

i. How the ball is caught

In most cases, catching the ball will result in the dismissal of a batsman. Several team members (S2, S4, S7, S8, S11) correctly interpreteed "catching the ball" as "intercepting the ball while it is still in the air", and modelled this relationship using a dependency. For instance, in her ISM, S7 used the definition: ball_caught is ball_intercepted && !ball_rolls;.

j. Modelling the transition between balls

In cricket, there is an interval between one ball and the next during which action on the part of players is suspended. For instance, the bowler may wish to review the location of the fielders before delivering the next ball. Whilst this review is in progress, it is not possible (for example) for the batsman to try to score additional runs. There is a significant point in the interval between one delivery and the next where the action associated with the first ball is deemed to be completed ("the ball is dead"). The subtlety of the "ball is dead" observable can be gauged from the fact that for the most part it is tacitly considered to be dead "when it is clear … that the fielding side and both batsmen … have ceased to regard it as in play". More often than not, this situation arises because the ball has been returned to the wicketkeepr or the bowler, a fact modelled by S1, S6, S10 and S11. Once the ball is declared dead, it may then have to be returned to the bowler for the next delivery. S2 and S9 modelled the general throwing of the ball from one fielder to another without reference to its final destination. This highlights the possibility of using the ISM to model informal activities that are incidental to the significant interpreted action in the cricket game.

k. Multiple batsmen and fielders

One might think that taking multiple agents on the cricket field into account would be a straightforward extension of an ISM. Despite this, only four individuals modelled multiple fielders or batsmen in the field. S3 modelled multiple fielders and a pair of batsmen in her model. S3, S7, and S9 observed that batsmen may or may not be preparing to hit the ball (be "on strike" or "off strike") according to which set of stumps they are located at and that they swap roles after scoring an odd number of runs. In addition, S9 recognised that there is a "next batsman" standing by off the field of play most of the time – an insight incorporated in the team ISM.

l. Scoring when the ball crosses the boundary

When the ball crosses the boundary of the field, the umpire may signal four runs, if the ball has touched the ground inside the boundary, or six, if the ball crosses the boundary whilst still in the air. S7 observed this signalling process and incorporated it in her ISM. Six out of the eleven team members took account of the ball crossing the boundary in their ISMs. They construed the scoring of

4 or 6 as occurring automatically when the ball crosses the boundary, and modelled it via an automated agent that monitors the ball movement.

m. Keeping score for multiple runs

Keeping score is straightforward for single runs, but becomes more complicated when multiple runs and other modes of scoring runs (such as accounting wides and no-balls – also called "extras") are taken into account. Most individuals (9 out of 11) considered simple score keeping based on single runs. Although both S9 and S6 have acknowledged the possibility of extras, only S6 has a set of observables which reflects a more sophisticated score keeping model. This allows him to record the extras and the total scores after multiple runs.

Contributions to the team's ISM

By comparing the individual ISMs to the team's ISM, it is not hard to figure out who made the most significant contribution to the team. For this purpose, the scripts of the individual ISMs were partitioned and compared under three categories initially, namely, observables, dependencies for enabling action, and triggered actions. As shown in Table 1, the contribution of each team member was awarded a numerical rating to reflect the degree of attention given to each scenario. Asterisks are used to indicate where a particularly unusual modelling contribution was made. Comparison was sometimes difficult since the same scenario can be modelled in various ways, e.g. solely by a set of dependencies or by observables complemented with triggered actions. The letters 'd' and 'p' are appended to entries in Table 1 to distinguish the use of dependencies from that of procedural actions. Individual perspectives that were adopted by the team are represented by shaded squares. In the case of some scenarios, most of the individuals shared a common perspective and the team simply integrated that into their ISM.

	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**S11**
Cricket Expert	-	-	-	-	Y	Y	-	-	-	Y	Y
Systems Analysis	-	Y	-	Y	-	-	Y	-	Y	-	-
Non-cricketer	-	-	Y	Y	-	-	Y	Y	Y	-	-
Technical Expert	-	-	Y	-	-	-	-	Y	Y	-	-
General Programmer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Background Research	Y	-	Y	-	Y	V	Y	-	-	Y	-

Table 2 – Distribution of expertise in Team A

The results of the analysis were correlated with information about the characteristics of the team members, as determined by a questionnaire conducted before any details of the exercise were known – see Table 2.

The results of the exercise seemed to indicate that the blending of individual ISMs into a team contribution is a much more straightforward process than combining individual traditional programs might have been. Even in this context, there was evidence that students used different strategies for observing events however, and this posed a challenge in integrating ISMs that most teams chose to address by selecting one or the other, rather than venturing to introduce dependencies to reconcile the

two approaches. The team report of Team A suggested that their integration strategy was to adopt one or two individual models as a base for the team submission and incorporate other perspectives into it. Although the team incorporated numerous features from individual ISMs, our analysis reveals that the team discarded certain details in the process. For instance, the team submission did not in general model a particular scenario in as much detail as it had been addressed in the best of the individual perspectives, as e.g. in scenarios (b), (g), (h), (i), (j), (k) and (l). Some scenarios that were developed by individual team members (e.g. in the case of (a), (c), (f) and (m)), were entirely omitted. This could have been the result of conflicts between perspectives that were difficult to reolve technically, or stem from group dynamics about which we have only very partial information.

An interesting, and somewhat unexpected, finding was that in some instances students with less sophisticated knowledge of cricket were the most successful in identifying the most appropriate way to model observation of the action in a cricket game. This could have been because the students with more experience of cricket found it harder to analyse their 'intuitive' knowledge. The exercise made use only of the rather primitive variants of the EM modelling tools that were available at the time. They did not allow the scope for concurrent distributed modelling that is afforded for instance by the web-enabled EDEN interpreter. It would be interesting to repeat this exercise making use of richer tool support in respect of distributed use and visualisation techniques that are now more readily accessible. There is also scope to blend the individual ISMs in other ways to drive alternative 'team' ISMs.

5.3 Other ISMs for cricket

To complete section 5, we shall briefly examine how similar principles apply to constructing ISMs based on other modes of observation of cricket, such as those identified in the introduction to section 5 above.

An effective abstract record of a game of cricket can be created by making a note of the outcome of every ball. If a batsman is deemed to be out as a result of a ball, this is registered as "the fall of a wicket", and if a batsman is given out as a direct consequence of playing the ball (rather than being 'run out'), this can be recorded as credited to the bowler. If the batsman is out as a result of a catch, the fielder responsible can also be recorded. If, on the other hand the batsman scores runs as a result of hitting the ball, these are credited to the batsman and added to his cumulative score. A ball may have no impact on the current score, as is the case when the batsman neither scores a run, nor is given out. There are also circumstances in which runs can be scored even though the batsman has not hit the ball. In this way, the progress of the batting team can be recorded in some detail as it develops, and at any given time can be summarised as "a certain number of runs scored, for the loss of a certain number of wickets".

In all formal cricket games, the current score is recorded in a special tabular form using a scorecard in which there is room to record the outcome of every ball, and to classify the runs scored according to which batsman scored them (if any), and the wickets according to the bowler and/or fielder to whom they should be credited. Summary information of this nature is made publicly available to the players and the spectators via a scoreboard.

The scorecard can be viewed as a form of model of a cricket game that is being elaborated incrementally as the game progresses. It is the kind of model that can be readily expressed using observables and dependencies. For instance, when a run is scored as a result of a batsman hitting the ball, this is recorded in the cumulative record of a batsman, and at one and the same time added to the current total that is recorded for the batsman (as typically displayed on the scoreboard) and to the current total score of the batting team. A simple way to represent this information in a definitive script is to build up a list of the runs credited to each batsman as they are being scored, to define the current score of the batsman as the sum of the entries in this list, and to define the current team score as the

sum of the runs accredited to all the batsman together with any runs that have been scored that cannot be credited to any batsman. An alternative form of ISM for a cricket game can be constructed in this way, as has been elaborated in the EDEN model developed in a final year undergraduate project in Computer Science by Pether, 2006.

It is self-evidently quite straightforward to link the cricket scoring ISM to the basic cricket ISM. The mode of observation employed in the basic cricket ISM is precisely such that the outcome of a ball, both in respect of runs and wickets and in respect of players to whom these should be credited, is unambiguously established. All that is required to combine the two ISMs is the introduction of suitable automatic actions (cf. section 5.1) to redefine the appropriate basic observables in the cricket scoring ISM, so that – for instance – when a ball results in runs being scored these are added to the list of runs credited to the batsman who hit the ball.

Observing a cricket match through its evolving scorecard – as is possible for instance through a textual electronic news medium – is the most abstract means of observation. The relationship between the formal account of a cricket match enshrined in a scorecard and what is informally observed by a spectator at the game exemplifies the kind of relationship between formal and informal perspectives that can be sustained in Empirical Modelling. What is actually observed is incomparably richer and more complex in nature than the scorecard can possibly reflect, yet the actual activity follows a pattern that is sufficiently stereotypical and ritualised that (possibly with the assistance of the umpire's human judgement in some aspects) it can be given an unambiguous interpretation.

Other kinds of account of a cricket game, such as a radio commentary, are also language based, but aim to capture more elements of the rich experience of being present at the game. One respect in which a commentary conveys more than a scorecard involves venturing to describe the precise locations of fielders. Rather than merely being given the information that a batsman has been given out caught, we may be told that the batsman was caught "on the square leg boundary". In this instance, the location of the fielder who makes the catch is only specified in a quite approximate manner, but can be realised in the imagination of the informed listener with some degree of objectivity. In this mode of communicating observation of a cricket game, there is nevertheless a pervasive problem of *indexicality* (cf. Rönkkö, 2007), whereby the same form of words ("the square leg boundary") may convey a different image to listeners according to their perception of their nterpretation in physical terms.

Another kind of ISM for cricket can potentially offer some help in resolving the communication problems associated with indexicality. In this model a redefinition of the form:

fielder/pos = positions/forward_short_leg/p

will locate a circle representing a fielder at a location corresponding to a plausible interpretation of the fielding position known as "forward short leg". There are dependencies in this ISM to reflect the fact that the location depends on the direction in which the ball is being bowled, and whether the batsman is right- or left-handed. The location of forward short leg may also be redefined to take account of variations in the interpretations of the qualifying concepts 'forward' and 'short', which may not be used entirely consistently by different commentators. In the fielding position ISM, redefinition of a suitable observable "short" serves to revise this interpretation, and will be reflected in the new position of the fielder on the display. The combination of textual and geometric information that is exercised in interacting with this ISM is suited to disambiguating the meanings of these terms in situated use.

ISMs for a cricket game can relate to observation of a form that is exceptionally difficult – indeed, arguably impossible – to express in purely logical terms (for an in-depth discussion of the distinction between the use of definitive scripts and logical specifications, see Beynon and Russ, 2008). Whilst

the examples of ISMs above have prominent formal symbolic aspects, relating to the laws of cricket, the system of scoring, and the conventional names of fielding positions, more characteristic applications of EM principles involve developing ISMs that reflect a more informal experiential emphasis (cf. section 3).

The key idea is that of initiating development from an informal empirical perspective and identifying patterns of interaction that follow stereotypical paths associated with thoughtfully engineered observables, dependencies and agency. In the ISMs considered so far, the emphasis is on the role that EM can play in enabling communication and negotiation of reality between designers, but other agent perspectives can also be significant in the comprehension process. One of the virtues of an EM approach is that models of processes and behaviours that are empirically established by focusing on modelling situations via definitive scripts and exercising agency within these situations via redefinition are simultaneously open to intervention from many different kinds of agent. This is in keeping with the way in which practice within a domain can – exceptionally – challenge the framework of even a well-established process.

A topical illustration can be drawn from the game of cricket itself. Many of the laws of cricket are contingent on whether the batsman adopts a right-handed or left-handed stance. Examples include the interpretation of the leg-before-wicket law, and the limitations that are placed on the location of fielders. In June 2008, the fact that Kevin Pietersen, an exceptionally gifted batsman, was able to exploit the practice of switching stance as the bowler was running up to bowl, obliged the cricketing authorities to reappraise the laws. Such a tactic was outside the scope of what had been considered plausible in drawing up the laws. This highlights the distinction between an approach to modelling that is predicated on a formal specification of the rules of behaviour and the EM approach. Whereas an action such as 'switching stance as the bowler runs up to bowl' could not be expressed in a model based strictly on the laws of cricket, it is readily possible to interpose such a redefinition into the ISM for basic cricket discussed in section 5.1 whilst it is being interpreted. If, further, the fielding positions were to be modelled using the dependencies expressed in the fielding position ISM, they would be instantaneously reconfigured accordingly.

The ISMs for cricket discussed above all draw in different ways on well-established accounts of a game of cricket that are essentially objective in character. A different approach to development is needed when creating an ISM from the empirical perspective. The objective is still to formulate a definitive script that can imitate the environment in which the individual agents (e.g. the players in a game of cricket) interact, but the initial focus is upon the typically subjective perceptions of the individual agents themselves. To this end, we can exploit an analysis of the perceptions of an agent based on classifying observables from an initially subjective perspective according to what observables the agent responds to, which are conditionally under the agent's control, and what perceived dependencies between observables are respected when an observable is change. A simulation of interaction between agents can be engineered in parallel with this analysis, which informs and is informed by the experience afforded by the emerging ISM.

The general principles and conceptual framework guiding such an approach is described in Adzhiev et al, 1994, though it has yet to be supported by mature EM tools. A cricket simulation was developed through analysing the perceptions of the player agents in this way in Nick Pownall's MSc dissertation (Pownall, 1993), but a detailed discussion of this ISM is beyond the scope of this paper.

A discussion of ISMs for cricket would be incomplete without some reference to the aspiration to model the game from the observational perspective of a physicist. This concept is developed to some degree – sufficient to give a plausible illusion of realism – in commercial computer games on the theme of cricket that are prominently advertised on the Web. The challenge of constructing a faithful physical model is illustrated by the fact that there is no entirely satisfactory physical explanation for

certain recognised features of actual play (such as the capacity that certain bowlers have for making the ball travel on a curved path to the batsman subject to conducive atmospheric conditions and to polishing the ball in an appropriate fashion). This limits the realism of models that are based on naïve theories of dynamics, and raises philosophical isssues that relate to the potential role of computer models in scientific experiment. An ISM that ventures a physically-based model of batting and bowling of this nature was developed by Mahmood in a final year undergraduate project (Mahmood, 2002). The extent to which EM affords a new conceptual framework for thinking about computer models in relation to scientific experiment is discussed in more detail in Beynon and Russ, 2008.

6 Summary

The main thrust of this paper has been to argue for a re-conceptualisation of computing that can give a better account of the development of digital artifacts that is required in giving support to distributed participatory design. The focus has been on describing the basic principles of Empirical Modelling as a means to developing construals in the form of interactive situation models. An ISM can reflect different ways in which a complex activity can be observed. The character of an ISM is such that different ISMs relating to the same situation can be refined and blended so as to support diverse modes of interaction and interpretation. Activity of this nature is crucial to supporting and sustaining the kind of design practice that is advocated by Löwgren and Stolterman (2004) in their account of "thoughtful interaction design".

Several different ISMs that have been developed as construals associated with the game of cricket have been used by way of illustration. The attentive reader will be in no doubt that the observational perspectives that are invoked by studying cricket are extremely rich, and comparable in complexity with the challenges of modelling business systems and processes. Evidence of the potential power of an EM approach can be adduced from the fact that the analysis of the collaborative development of a basic cricket ISM by a team of students, as described in section 5.2, can give plausible insight into the way in which they were thinking about such a wide range of subtle issues concerning agency, interpretation and interaction.

Further tool development, together with empirical studies of use on a broader scale that can only be achieved through wider adoption, are crucial to realising the full potential and gaining a better understanding of the limitations of the approach. The diverse ISMs for cricket discussed in this paper hint at many ways in which EM can support design as collaborative knowledge construction in the spirit of Löwgren and Stolterman (2004). The scope for combining ISMs to create richer and more expressive digital artefacts contrasts with the difficulties of integrating different formal methods of specification of systems, identified as a 'Grand Challenge' by Dines Bjørner (2003).

Acknowledgments

We are indebted to the many developers of tools and models that have been mentioned in this paper.

References

Beynon, W.M., Adzhiev, V.D., Cartwright, A.J. and Yung, Y.P., "A New Computer-Based Tool for Conceptual Design," Proc. Computer Tools for Conceptual Design, Univ. of Lancaster, 171-188, 1994.

Beynon, W.M., Adzhiev, V.D., Cartwright, A.J. and Yung, Y.P., "A Computational Model for Multiagent Interaction in Concurrent Engineering," Proc. CEEDA'94, Bournemouth Univ., 227-232, 1994.

Beynon, W.M. and Sun, P-H., "Computer-mediated communication: a Distributed Empirical Modelling perspective." Proceedings of CT'99, San Francisco, August 1999.

Beynon, W.M., Rungrattanaubol, J. and Sinclair, J., "Formal Specification from an Observation-Oriented Perspective," *Journal of Universal Computer Science*, Vol. 6 (4), 407-421, 2000.

Beynon, W.M., "Radical Empiricism, Empirical Modelling and the nature of knowing," In *Cognitive Technologies and the Pragmatics of Cognition: Special Issue of Pragmatics and Cognition* (ed. Itiel E Dror), 13:3, December 2005, 615-646.

Beynon, W.M., Russ, S.B. and McCarty, W.F., "Human Computing: Modelling with Meaning," *Literary and Linguistic Computing* 21(2), 2006, 141-157.

Beynon, W.M., Boyatt, R.C. and Russ, S.B., "Rethinking programming," Proc. IEEE 3rd Int Conference in Information Technology: New Generations, 2006, 149-154.

Beynon, W.M., and Chan, Z.E., "A conception of computing technology better suited to distributed participatory design," in Proc. NordiCHI Workshop on Distributed Participatory Design, Oslo, Norway, October 2006.

Beynon, W.M., "Computing technology for learning - in need of a radical new conception," In *Journal of Educational Technology & Society*, 10 (1), 2007, 94-106.

Beynon, W.M. and Russ, S.B., "Experimenting with Computing", *Journal of Applied Logic: Special Issue on the Philosophy of Computer Science*, 2008, at the url: http://dx.doi.org/10.1016/j.jal.2008.09.008.

Bjørner, D., "A 'Grand Challenge': Unifying Theories of Formal Methods – Integrating Formal Methods," http://www2.imm.dtu.dk/~db/formal-methods/challenges/utopia/ (accessed 07/09/08)

Bodker, S., Ehn, P., Kammersgaard, J., Kyng, M. and Sundblad, Y., "A UTOPIAN experience: On design of powerful computer-based tools for skilled graphic workers." In *Computers and Democracy – a Scandinavian Challenge*. G. Bjerknes, P. Ehn, and M. Kyng, Eds. Aldershot, Gower, Avebury, England, 1987, 251-278.

Bornat, R., "Is 'Computer Science' science?," In Proc ECAP 2006.

Crabtree, A., *Designing Collaborative Systems: A Practical Guide to Ethnography*, Springer, 2003.

Danielsson, K. "The shift from user, to learner, to participant: An inevitable development or (just a) mere coincidence?" Proceedings of the

Participatory Design Conference, volume II, 49-52, Toronto, Canada, 2004.

D'Ornellas, H.P. "Agent Oriented Modelling for Collaborative Group Learning," MSc Project Report, Department of Computer Science, University of Warwick, 1998.

Farshchian, B.A. and Divitini, M. "Using email and WWW in a distributed participatory design project." SIGGROUP Bull. 20, 1 (Apr. 1999), 1999, 10-15.

C. Floyd. "Software development as reality construction," In C. Floyd, H. Züllighoven, R. Budde, and R. Keil-Slawik (eds.), *Software Development and Reality Construction*. Springer Verlag, Berlin, 1992.

Friis, S. Action research on systems development: case study of changing actor roles. SIGCAS Comput. Soc. 18, 1, Jan. 1988, 22-31

James, W. *Essays in Radical Empiricism*, Bison Books, 1996.

Jullien, F. *The Propensity of Things: Towards a History of Efficacy in* China (trans. Janet Lloyd), Zone Books, New York, 1995.

Löwgren, J. and Stolterman, E. *Thoughtful Interaction Design: A Design Perspective on Information Technology*, The MIT Press, 2004

Mahmood, N. "An EM cricket simulation". EM model and screenshot at url:

http://empublic.dcs.warwick.ac.uk/projects/semcricMahmood2002/

Pether, C. "An EM model for cricket scoring". EM model at url:

http://empublic.dcs.warwick.ac.uk/projects/cricketscoringPether2006/

Pownall, N. "A agent-oriented cricket simulation". EM model at url:

http://empublic.dcs.warwick.ac.uk/projects/cricsimPownall1993/

Kari Rönkkö. Interpretation, interaction and reality construction in software engineering: An explanatory model. Information and Software Technology, 49:682–693, 2007.

Sheth, C. "An Investigation into the application of the Distributed Definitive Programming Paradigm in a Teaching Environment: The Development of a Virtual Electrical Laboratory," MSc Project Report, Department of Computer Science, University of Warwick, 1998.

Strauss, A. and Corbin, J.M., Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. 1998. Springer, Berlin.

Woodbury, M. "If it doesn't have a wicket, it just ain't cricket," http://media.www.jhunewsletter.com/media/storage/paper932/news/2002/11/15/Features/If.It.Doesnt.Have.A.Wicket.It.Just.Aint.Cricket-2249081.shtml (accessed 06/09/08)

Hopps, D and Wilson, A. "Pietersen's tactic forces MCC into emergency debate", The Guardian newspaper online at the url (accessed 9/10/08):

http://www.guardian.co.uk/sport/2008/jun/17/cricket.englandcricketteam

Empirical Modelling homepage: http://www.dcs.warwick.ac.uk/modelling/ (accessed 07/09/08)

Empirical Modelling archive: http://empublic.dcs.warwick.ac.uk/projects/ (accessed 07/09/08)

The Web-EDEN interpreter: http://weden.dcs.warwick.ac.uk/

(accessed 07/09/08)

The Laws of Cricket: http://en.wikipedia.org/wiki/Laws_of_cricket

(accessed 17/06/09)