6. Conclusions and Future Work

6.1 Conclusions

The aim of this research is to develop a component middleware technology for mobile computing devices in the discovery and capture of contextual information, using the situated reasoning concept of constructive memory.

The emerging research area of context-aware pervasive computing is envisioned to significantly increase the performance of applications operating in networked and sensor-based contexts. With the advent of the context-aware computing paradigm, adaptation of these applications to various situations of interest will become feasible. This in turn, will improve the applications' performance and the quality of services they provide to users. Context-aware applications rely on the existence of sensors, devices and communication facilities in pervasive network settings, and are dependent on the type and characteristics of context they attempt to reason about. Consequently, different challenges that originate from these dependencies currently exist and impede the proliferation of context-aware applications. This research has developed a modified architecture in an attempt to deal with some these challenges associated with context-aware and pervasive computing. It is based on the context theory for achieving reasoning, verification and prediction of context in context-aware, pervasive environments. With this modified architecture, this research has examined the impact and the process of reasoning about contextual information, using a constructive memory model to predict contexts.

Another problem that has existed in context-aware systems is the training required to map all possible contextual information and context-type combinations. These methods of training context-aware systems can be a long and arduous process and they are ill-equipped to address the difficulties associated in developing future context-aware computing applications. This research has quantified *long and arduous* as the mapping required between the number of contexts and the number of device and sensor combinations per context to be trained by the context-aware application. The conceptual framework in this thesis addresses this challenge in context awareness by incorporating a constructive memory model. The operational characteristics of the memory model implemented are based on the theory of constructive memory. What is constructed by the system is a collection of experiences that predict the current context, based on the contextual information the system has stored already. The conceptual framework developed differs from previous context-aware architectures by implementing a constructive memory query layer. This layer is designed to induce and construct potential contexts, based on the contextual information that is discovered, and recalling previous experiences where such information has been seen in specific contexts. This process only requires a small amount of learning and interaction before it is able to predict a context based on the memory-matching algorithm implemented and demonstrated through performance studies in this thesis. Additionally, the framework is founded on a set of well-integrated abstractions that support the developer through analysis, design, implementation and beyond. Specifically, the context-aware and memory model used for analysis and design tasks maps to a context management infrastructure that is exploited by the implementation. Overall, the constructive memory model provides a conceptual framework that demonstrates its applicability in context-aware systems.

The notion of constructive memory contradicts many views of knowledge being unrelated to either its locus or application. Memory in this thesis is modelled as a dynamic process, based on the model of constructive memory. It is not a static imprint to be stored in a specific location and retrieved for use later. The memory construction processes are not predefined but are influenced by contextual information received at the time. Memories that are constructed may not match the original experience exactly as it was first experienced, but change according to when, where and with what the memory system is cued. Experience is defined as the accumulations of knowledge or skill that results from direct participation in events or activities. Learning is a process whereby knowledge is created through the transformation of experience. Experience plays a key role in the learning process. A broad spectrum of learning approaches has been developed, but essentially learning takes place when the system interacts with, or is stimulated by an environment. These behaviours of the memory system are dependent on the current environment, the current contextual information being discovered and the interactions between this information and the environment.

The research presented in this thesis provides a perspective of constructive memory within the domain of context awareness that allows a computational memory model to be developed. A computational framework for constructing context has been designed and implemented to demonstrate the various behaviours of the constructive memory system not easily producible by standard context information retrieval systems. The system organises its experience to construct a meaning for contexts. The system incrementally learns as there are more experiences stored for a given context, based on application interactions. A constructive memory model is used to construct a context, which enables the application to anticipate future contexts and go beyond the interpreted contextual information. Concept formation takes account of context and the experience, as well as interactions between them. A concept is learned in three ways: memory activation, memory reactivation and memory construction. In memory activation, the concept is formed from the activated experiences. Memory reactivation occurs when the initially formed concept fails in the validation process. A memory can be reactivated to form a new concept when a memory cue is able to be identified in the context. In memory construction, a concept is learned when the application's contextual knowledge repository is used to induce a context.

This thesis and framework make a number of important contributions. Chapter 2 was motivated by the need for a clear distinction between context information and the more nebulous concept of context, and argued that understandings of the characteristics of the former are the key to the design of successful context-aware systems. With this in mind, the chapter presented a novel characterisation of context information in terms of its source, temporal and quality characteristics, and dependencies. The requirements arising from this characterisation formed the foundations of the proposed framework, directly informing the design of the constructive memory model approach described in the conceptual framework. Chapter 3 served to integrate the theoretical contributions from previous literature into a system architecture incorporating the discovery, management and constructive memory of context information. The architecture addresses the key challenges of pervasive systems in relation to autonomy, resource limitations, scalability and dynamic aspects of computing environments and user requirements. Chapter 4 presented the architecture implementation in the form of a J2ME application. The application incorporated Bluetooth discovery as the context discovery and reception layers, an RMS as the context management and repository layer and a constructive memory-matching algorithm as the constructive memory query layer. This prototype is valuable both as a form of validation for the design of the architecture, and as a test bed for the underlying theoretical foundations. Chapter 5 provided results and an evaluation of a series of experiments conducted as the basis for the research claims. The results demonstrated that only after a small amount of learning and interaction, the application was able to learn and construct from memory a predicted context based on the current devices discovered and what the systems had seen in previous experiences. The results obtained are encouraging and would be further strengthened by additional experiments involving a variety of context and application domains. Nevertheless, this is an area of future work and directions for the research.

The overall approach to the research and the process in which constructive memory is incorporated in context-aware systems is best summarised by the reiterating the statement:

Constructed Context $= \sum$ What I see now + [What I have seen in relation to what I see now];

Where *What I see now* are the current devices and sensors being discovered and *What I have seen in relation to what I see now* are the experiences during which the current devices being discovered are stored and saved by the application. The constructive memory algorithm uses these experiences to construct contexts and inductively learn from previous experiences.

6.2 Future Work and Directions

The research presented is intended as a working prototype implementation, both motivating and illustrating how a constructive memory model can be applied to context-aware systems. Since the area of context awareness is very broad in the computer science domain, there are many opportunities for future work and directions can be explored further.

The crucial role of appropriate abstractions and programming models in the success and adoption of context-aware applications has long been recognised. This view has been reiterated several times since and is backed by a wider recognition of the inadequacy of current programming techniques for pervasive computing software in general. However, the literature review illustrated that very little progress has been made in this area. While significant attention has been devoted to the development of context discovery infrastructures, the programming models used in conjunction with these remain inadequate and often oversimplified.

Another area of future work is the growing awareness of common usability problems associated with context-aware software. These pose significant design challenges. Perhaps the largest challenge lies in exploiting context information such that applications are highly autonomous, requiring little user input, while at the same time ensuring that control ultimately lies with the user. In order to achieve this balance, there is a need for transparency, such that the actions of applications are visible to users and appear consistent and predictable, as well as feedback mechanisms that allow users to override any inappropriate actions. Personalisation is also a key consideration in ensuring that context-aware applications are acceptable to users; that is, applications developers must recognise that users have differing requirements and that these may evolve over time.

An essential topic for future work is that of context management. Usability problems arising in context-aware software as a result of privacy concerns and imperfect contextual information are well recognised. These can be alleviated in part by the use of appropriate management infrastructure that implements privacy policies and performs tasks such as conflict detection and resolution, and enforcement of integrity constraints. Such mechanisms are already in place in some location management systems, but have not yet been adequately generalised to deal with broader and richer types of context information. This represents a difficult task, owing to the extremely heterogeneous types of context information present in many context-aware systems.

The literature review in Chapter 2 demonstrated that the design process associated with context-aware software is poorly understood, and that no mature design tools are available. The research presented in this thesis has partially addressed this issue by providing a conceptual framework that can be used as a basis for the exploration and specification of an application's context requirements. However, there is also a requirement for interaction design techniques and tools that facilitate the design of user interfaces that address the various human-computer interaction challenges of pervasive computing environments, and overcome known usability problems introduced by context awareness in relation to transparency, consistency, predictability, privacy and user control. In addition, better understanding of the overall software engineering process associated with context-aware applications is required. This process remains to be explored in a more general sense and then formalised as a well-defined software engineering methodology.

Further evaluation of context-aware software in realistic settings is required. This is currently difficult, as seamless, large scale pervasive computing has not arrived, and easy to use *plug-and-play* sensing components are not yet available for discovering rich types of contextual information. Empirical evaluation of context-aware software is also important for several reasons. First, it is crucial in identifying new usability problems, and in exploring inappropriate solutions to these. Second, it provides important validation and feedback in terms of the suitability of proposed design, context-modelling and programming techniques. Finally, it will eventually lead researchers to better understand the most compelling uses of context awareness, enabling the development of applications that are capable of achieving widespread acceptance and use.

Finally, this thesis, as stated a number of times has defined context to mean *place* and has not used people, tasks or activities in determining context. An area that can be further explored would be to include all aspects of context, using the constructive model for context-aware applications. This area of research potentially might put forward a claim that such decision support tools, such as the constructive memory model implemented in this research, are something that context-aware system developers will consider and include in the future.