Submission to the Australasian Computing Doctoral Consortium

Carl Schultz
Affiliation: Department of Computer Science, The University of Auckland
Level 3 Room 384, Science Centre (Building 303)
38 Princes Street, Auckland, New Zealand
Email: csch050@aucklanduni.ac.nz
Phone: (9) 373 7599 ext. 87458
Enrolment: March 2006
Thesis title: **A Framework for the Application of Qualitative Spatial and Temporal Reasoning**

Keywords: qualitative spatial temporal reasoning, software engineering, knowledge engineering
Key references: ***(NB: I can provide copies of articles [6], [7] and [8] if required)***

[1] Cohn A, Renz J, Qualitative spatial reasoning. In van Harmelen F, Lifschitz V, Porter B (eds): Handbook of Knowledge Representation; Elsevier Science, 2007.
[2] Allen JF, Maintaining knowledge about temporal intervals. Communications of the ACM; 1983, 26 (11), pp.832-843.
[3] Dylla F, Frommberger L, Wallgrün JO, Wolter D, SparQ: A toolbox for qualitative spatial representation and reasoning. In Proceedings of the Workshop on Qualitative Constraint Calculi: Application and Integration, 2006.
[4] Wölfl S, Mossakowski T, Schröder L, Qualitative constraint calculi: Heterogeneous verification of composition tables. In Proceedings of the Twentieth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2007); AAAI Press, 2007, pp. 665-670.
[5] Schultz CPL, Clephane TR, Guesgen HW, Amor R, Utilization of qualitative spatial reasoning in geographic information systems. In Riedl A, Kainz W, Elmes GA (eds): 6th International Symposium on Spatial Data Handling: Progress in Spatial Data Handling; Springer-Verlag, Berlin Heidelberg, 2006, 12, pp. 27-42.
[6] Schultz CPL, Amor R, Guesgen HW, Towards a theory of application for QSTR systems. To appear in AAAI Symposium: Benchmarking of Qualitative Spatial and Temporal Reasoning Systems, March 2009.
[7] Schultz CPL, Amor R, Lobb B, Guesgen HW, Qualitative design support for engineering and architecture. To appear in Advanced Engineering Informatics; Elsevier, 2009, 23(1) pp. 68-80.
[8] Schultz CPL, Amor R, Guesgen HW, Unit Testing for Qualitative Spatial and Temporal Reasoning. Submitted to Florida Artificial Intelligence Research Society Conference (FLAIRS 2009).

Background and Research Problem

Over the last 25 years, researchers in artificial intelligence (AI) have developed a variety of sophisticated methods for qualitative spatial and temporal reasoning (QSTR) [1] that provide flexible querying and reasoning for vague and uncertain spatial and temporal data. The seminal example of QSTR is Allen's interval calculus [2] which defines a set of thirteen atomic relations between time intervals (refer Figure 1), and an algorithm for reasoning about networks of temporal relations; given qualitative relations between intervals t_1 and t_2, and intervals t_2 and t_3, we can infer the possible relations between intervals t_1 and t_3 e.g.

If t_1 is before t_2 **and** t_2 contains t_3 **then** t_1 is before t_3
If t_1 overlaps t_2 **and** t_2 during t_3 **then** t_1 (overlaps, or is during, or starts) t_3

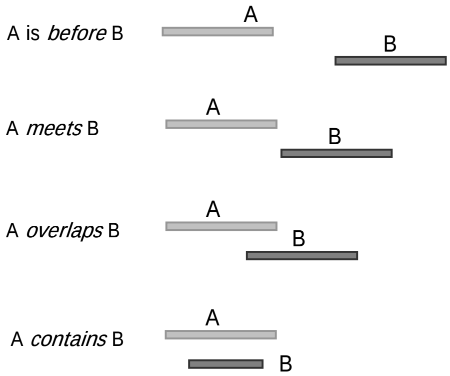

Fig 1. An extract of Allen's [1] qualitative relations between temporal intervals.

Fig 2. Screenshot of transparency (top) used to visualise results of a qualitative query (bottom) in TreeSap [5].

Despite significant theoretical advances, there is a distinct absence of applications that make use of these techniques. This lack of interest outside the community has been acknowledged as a growing concern in prominent QSTR tracks at recent major AI conferences, particularly because spatial and temporal information are a central component in a wide range of disciplines such as geographic information systems (GIS), robotics, scheduling, and construction IT [1].

My PhD project addresses this problem by aiming to establish a framework of methodologies that support the development of QSTR applications. My view is that the QSTR community has not adequately addressed or acknowledged the unique issues of designing and developing QSTR applications compared to more traditional systems. Standard research focuses on the theoretical analysis of QSTR systems, most recently [3,4]. Alternatively, we are taking a significantly novel approach by considering the problem from the perspective of the application designer, drawing from results in software engineering and knowledge engineering.

<u>Methodology for Developing QSTR Application Framework</u>

My project consists of the following tasks for developing the proposed framework.

I began by undertaking five application based case studies covering project management, robotics, astronomy, geographic information systems, and construction IT. Each case study involved a domain specific problem to which I attempted to apply a QSTR solution. The aim was to directly encounter any issues that arise when attempting to design and implement QSTR models for solving particular problems. Patterns identified during the case studies and other literature review based work are being used to develop and refine the application framework. Note that the case studies are not part of the framework itself; I am using the case studies to highlight issues relating to the application of QSTR, as a guide to developing the framework.

1 For example, next March a symposium track in the Association for the Advancement of AI (AAAI) called "Benchmarking of Qualitative Spatial and Temporal Reasoning Systems" is being run to "push the development of qualitative reasoning methods and systems towards application-relevant problems".

The next task was to establish framework components that are responsible for different aspects of application development. The major framework components are derived from phases in software development processes such as design, implementation, and validation. These major components are now being developed further by analysing relevant sub-areas. For example, sub-areas of "validation" include unit testing, integration testing, test coverage, confidence, performance metrics and so on. Sub-areas have been inspired by analogous research in standard QSTR literature, software engineering and knowledge engineering.

The final phase is framework validation, i.e. ensuring that the components in the framework, and the relationships between components, are effective for supporting the application of QSTR. For this we are considering revisiting case studies to give concrete examples of component use, and running a study in which a developer with no prior QSTR experience will use the framework to produce a QSTR application.

Preliminary Results for the QSTR Application Framework

I have conducted five case studies, identified the major framework components and their basic relationships, and have developed a number of sub-areas within each major component.

One case study is in the area of Geographic Information Systems (GIS). Modern GIS require tools for manipulating, viewing and flexible querying of geographic information. We have developed a system called TreeSap GIS [5] that explores the use of QSTR, and demonstrates its applicability towards more sophisticated, yet widely accessible, qualitative query support, as illustrated in Figure 2.

The major components of our framework are design, implementation and validation. It must be noted that the framework is not a software tool and does not contain code libraries. It is a theory of the application of QSTR, developed from first-principles by redefining the underlying set theoretic definition for QSTR [6] (refer to section "QSTR Application Theory") according to our novel perspective of the application designer. We use this new definition to derive principles [7] about

- qualitative models (what they are, how they relate to other models such as numerical or fuzzy ones, how they should be designed to meet specific task requirements, how to integrate existing models such as Allen's interval calculus [1], and how to quickly explore different design options for ambiguous cases [6] (refer to section "Modelling Ambiguity")),
- qualitative reasoning (how it operates, how it can be used to address a problem given the available domain knowledge, and how domain knowledge can be formalised by applying a useful construct that we have developed [8] (refer to section "Critical Boundaries for Unit Testing", paragraph 3)),
- tasks that can be performed by a qualitative system (for example, we show that all basic QSTR tasks, including inference and consistency tests, are a sequence of queries and insertions, and so determining the logical limits of querying in a particular QSTR system is fundamental in analysing its capabilities, behaviours and complexity; moreover we are developing methods that can precisely calculate these query limits), and

- higher level QSTR application characteristics (for example, designing rapidly changing models such as a mobile robot's local environment model, compared to static models such as the GIS case above)

These extend to principles and methodologies for implementing and validating QSTR applications [7,8]. The framework makes no assumptions about the ordering of the processes or the software lifecycle model employed.

Areas in Which Expert Advice is Sought

Two immediate problems are (a) validating the framework, and (b) finding a suitable balance between breadth and depth of areas developed in the framework. Any advice or comments about these issues would be greatly appreciated.

Validation

I am not clear on how to best convince a reviewer that a developer will be more productive by using my framework. Unfortunately, due to the very lack of QSTR applications, there are no standards that could be used for comparison (e.g. there is no definition of a useful or correct QSTR application, and no prominent examples). Furthermore, not only do I need methods for validating each individual component, but also for validating the integration of the components (i.e. the framework as a whole).

We are considering three methods for validation. Firstly, I am revisiting the case studies (along with a number of other existing QSTR applications) to illustrate exactly how the components of the framework are used by a developer. My aim is to establish that the framework does indeed facilitate different types of QSTR systems across a range of disciplines.

Secondly, we are looking to conduct a study involving a software developer with no previous experience of QSTR (e.g. as a student project). The participant will be asked to use the framework to develop a QSTR based software tool that addresses a given domain specific problem, and then report on the results. This feedback can then be used to refine the framework and support the argument that the framework is useful.

Thirdly, I am trying to emphasise the difference between approaching a problem with and without the framework. For example, one particular verification component prescribes critical boundaries for unit testing (analogous to boundary testing in normal programming), without which a developer is faced with an infinite number of possible test cases and no basis for test selection.

Breadth vs. Depth

The project covers a wide range of topics. My concern is that, if I do not analyse the individual components in enough detail, then regardless of how well they work together, a reader will not find the framework very satisfying. On the other hand, too much detailed analysis of individual components will obscure the overall framework structure.

In the dissertation, should more focus be given to showing the relationship between components (e.g. how components interact and support each other), or analysing components in isolation but with greater detail? Are there any suggestions or guidelines that I could consider for getting a good balance?