

Introducing Reflective Techniques to Message Hierarchies

This research is being conducted within the scope of the Virtual Logistics multi-Agent Brokerage (V-LAB) research project at the Department of Information Technology, National University of Ireland, Galway. The goal of this research is to define enhancements for Enterprise messaging systems allowing them to be more adaptable to their runtime environment. Research on enhancements for enterprise-messaging systems focuses on hierarchical channels or topics within publish/subscribe systems. Hierarchical channels are used for routing situations that are more or less static. Hierarchical channels require that the channel namespace schema be both well defined and universally understood. In large-scale systems, grouping events into related types (i.e. channels) helps to manage large volumes of different events [1].

The responsibility for choosing a channel in which to publish is left to the publishing application. The *channel selection logic* is integrated in the publisher's application code. This results in a tight coupling between the channel hierarchy and publisher's code, making changes in the hierarchy difficult and restricting the ability of the hierarchy to evolve and change. Any alterations must be propagated to the publisher's application code. Due to this tight coupling it is important to define accurate and correct channel hierarchies early in the development process. Any changes required to the hierarchy downstream will result in a high cost of change; once a system is deployed it becomes problematic to change the hierarchy.

Reflection is currently a hot research topic within software engineering and development. A common definition of reflection is "A reflective system is one that provides a representation of its own behaviour which is amenable to inspection and adaptation, and is causally connected to the underlying behaviour it describes"[2]. Schmidt advocates the use of reflection within middleware for advanced adaptive behaviour "The reflective middleware model is a principled and efficient way of dealing with highly dynamic environments yet supports development of flexible and adaptive systems and applications"[3] "This reflective flexibility diminishes the importance of many initial design decisions by offering late- and runtime-binding options to accommodate actual operating environments at the time of deployment, instead of only anticipated operating environments at design time."[4]

Researchers at the IBM T.J Watson research centre in New York have taken a unique approach with application message conditions. They have identified that message receipt and processing by final recipients are often important criteria that represent a condition on further processing by the sender [5]. At present no defined middleware support exists to aid in the development of applications that require the management of conditions on messages. Their solution uniquely shifts the responsibilities for implementing the management of conditions on messages from the application to the middleware.

By applying a shift in responsibility of the channel selection logic from the publishing client to the middleware we allow the application of reflective techniques in channel hierarchies. This shift in responsibility to the service allows more control over the definition, creation and maintenance of channel/topic hierarchies. The enhanced service

is now able to apply reflective and adaptive techniques to dynamically grow and improve its hierarchy to meet the needs of its changing environment and operating conditions. The resulting system is analogous to a post office: messages are placed into a black box, how the messages are sorted is of little concern to the sender.

This work forms part of the V-LAB research project whose goal is to develop a system for the virtual brokerage, optimisation and management of road freight carriers. V-LAB combines diverse technologies to create a multi-agent software system that will facilitate companies competing for extra work convenient to their individual schedule. The system will be a real time broker that offers parcels up for delivery, and then allows companies to compete for these jobs by creating tenders for their delivery. The purpose of this is to utilise empty vehicle capacity, therefore increasing cost efficiency.

One aspect of messaging within V-LAB deals with job requests for the brokerage. A jobs location and completion time is an important factor for companies competing in this marketplace. However, depending on the scenario of the deployment, the scales used to measure the location and times differ. In a continental deployment the country of origin and destination are of high priority to shippers, as not every company will service all the potential countries and regions. Completion times are measured in days and weeks. With a metropolitan scale deployment the source and destination become less important as it is more likely that any city courier will service the entire city, criteria such as package size and speed of delivery become more important, with completion times in hours.

Other deployment scenarios with different domain specific criteria include fruit, furniture, money, dangerous material, liquid, oil, gas, cars, etc. A static channel hierarchy requires predefined criteria for every potential deployment scenario in advance. Utilising a reflective channel hierarchy approach lets the consumers choose the criteria, allowing a channel hierarchy to evolve (or grow) and constantly adapt into one customised specifically for the deployment environment. Starting from a single channel (seed) a customised channel hierarchy (tree) can grow based on the expressed requirements of its consumers.

References

- [1] P. R. Pietzuch and J. M. Bacon, "Hermes: A Distributed Event-Based Middleware Architecture," 2002.
- [2] G. Coulson, "What is Reflective Middleware?," *IEEE Distributed Systems Online*, 2002.
- [3] F. Kon, F. Costa, G. Blair, and R. H. Campbell, "The Case for Reflective Middleware," *Communications of the ACM*, vol. 45, 2002.
- [4] R. E. Schantz and D. C. Schmidt, "Middleware for Distributed Systems: Evolving the Common Structure for Network-centric Applications," in *Encyclopedia of Software Engineering*: Wiley & Sons, 2001.
- [5] S. Tai, T. Mikalsen, I. Rouvellou, and S. M. S. Jr, "Conditional Messaging: Extending Reliable Messaging with Application Conditions," presented at 22nd International Conference on Distributed Computing Systems, Vienna, Austria, 2002.