Middleware's Presentism: Asynchrony, Flow, Finance, and the Enterprise
HaPoP 2018 (Oxford, UK) Submission

Michael Castelle
Assistant Professor, University of Warwick
M.Castelle.1@warwick.ac.uk +44 7506919298

[NOTE: A variant of this abstract was previously submitted for the Fall 2017 History and Philosophy of Computing (HaPoC) conference in Brno, but I was unable to attend due to my starting a new position at the same time as the conference. I had a short email discussion with HaPoP co-organizer Tomas Petricek and he gave me permission to submit an updated/edited version for HaPoP 2018.]

This paper will examine the primary intellectual controversies in the early development of distributed computing systems, which isolated the distinctive features of nascent inter-process and inter-computer communication (e.g. synchronous vs. asynchronous (Liskov, 1979); procedure-oriented vs. message-oriented (Lauer & Needham, 1978); constraints on temporal and causal ordering (Lamport, 1978; Cheriton & Skeen, 1993)), and argue that these debates, interwoven with their eventual implementation and commercialization for financial services and manufacturing applications (and beyond), constitute a form of pragmatic, materialist, applied philosophy on topics of communication, temporality, and causality. Further, by specifically examining the various systems and distributed programming environments developed at Xerox PARC, Stanford and Cambridge in the late 1970s through the late 1980s—including Remote Procedure Call (Nelson, 1981), the V Kernel (Cheriton, 1984), ISIS (Birman et. al., 1985), and The Information Bus (Skeen, 1992)—I will show how the distinct communicative styles engendered by this research initially came to be valued both by Wall Street brokerage firms increasingly overwhelmed by a surfeit of heterogeneous, incompatible digital data feeds and by a variety of organizations with ossifying 'stovepipe' legacy infrastructure. I will in turn suggest that these systems, which emphasize asynchronous messaging and broadcast communication, bear a conceptually (and arguably ontologically) dual relationship to systems programming techniques which (deliberately or unconsciously) strove to deny qualities of (and concerns for) processuality, temporality, and the material unpredictability of failure.

Inspired by these techniques, the industry which subsequently emerged across the 1990s—that of *message-oriented middleware* (Horwitt, 1993; Banavar et. al., 1999)—was in part predicated on the increasing interest of traders in harnessing and differentially attending to real-time flows of securities information and news (Ranadive, 1999), but also in part predicated on the related need of many large organizations to bridge a new generation of client/server architectures and desktop workstations with legacy mainframe systems, batch-oriented applications, and real-time systems (Schulte, 1996). The distinctive *publish/subscribe* communication pattern which emerged in these message-oriented systems (Cheriton & Zwaenepoel, 1985; Oki et. al., 1993) was one in which conceptually centralized (if logically distributed) flows of messages would be 'published', for which 'subscribers' (e.g. to subjects like '*.ibm.news.reuters') would be asynchronously notified when events of interest (e.g. Reuters headlines about IBM) occurred. This publish/subscribe paradigm would re-emerge in different asynchronous/distributed contexts over the following decades (Eugster et. al., 2003), from "push media" (Kelly & Wolf, 1997) to the Internet of Things (Stanford-Clark, 2002).

As mentioned above, I will argue that these technologies and techniques which prioritize *data-in-motion* can be seen to be in a dualistic relationship with data traditions of a 'set-theoretic Platonism' (Mac Lane, 1986) such as the relational database model (Codd, 1970), which tend towards a static world devoid of asynchronous events. In this dichotomy between the atemporal data of the archive and the highly temporal data streams of messaging middleware—what functional programming researchers would call *codata* (Turner, 2004)—we can see fascinating parallels to long-running philosophical debates between a deictic and a 'tenseless' view of time inspired by McTaggart (1921). The contemporary message-oriented middleware underlying our largest digital media platforms, then, can be seen as unconsciously in the tradition of G.H. Mead: a "world of events" for which sociality "is a process continually passing into the future" and in which "objects exist in nature as the patterns of our actions" (Mead, 1932, p. 190).

Bibliography

Banavar, G., Chandra, T., Strom, R., & Sturman, D. (1999). A Case for Message Oriented Middleware. In P. Jayanti (Ed.), *Distributed Computing* (pp. 1–17). Springer Berlin Heidelberg.

Birman, K., El Abbadi, A., Dietrich, W. C., Joseph, T., & Raeuchle, T. (1985, January). An Overview of the ISIS Project. IEEE Distributed Processing Technical Committee Newsletter.

Cheriton, D. R. (1984). The V Kernel: A Software Base for Distributed Systems. *IEEE Softw.*, *1*(2), 19–42.

Cheriton, D. R., & Skeen, D. (1993). Understanding the Limitations of Causally and Totally Ordered Communication. In *Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles* (pp. 44–57). New York, NY, USA: ACM.

Cheriton, D. R., & Zwaenepoel, W. (1985). Distributed Process Groups in the V Kernel. *ACM Transactions on Computer Systems*, *3*(2), 77–107.

Codd, E. F. (1970). A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM*, *13*(6), 377–387.

Eugster, P. T., Felber, P. A., Guerraoui, R., & Kermarrec, A.-M. (2003). The Many Faces of Publish/Subscribe. Presented at the ACM Computing Surveys.

Horwitt, E. (1993). Vendors get caught up in middle (ware). *Computerworld*.

Kelly, K., & Wolf, G. (1997, March). PUSH! Retrieved June 1, 2017, from https://www.wired.com/1997/03/ff-push/

Lamport, L. (1978). Time, Clocks, and the Ordering of Events in a Distributed System. *Commun. ACM*, *21*(7), 558–565.

Lauer, H. C., & Needham, R. M. (1978). On the Duality of Operating Systems Structures. In *Proc. Second International Symposium on Operating Systems, IR1A*.

Liskov, B. (1979). Primitives for Distributed Computing. In *Proceedings of the Seventh ACM Symposium on Operating Systems Principles* (pp. 33–42). New York, NY, USA: ACM.

Mac Lane, S. (1986). *Mathematics Form and Function*. Springer-Verlag.

McTaggart, J. M. E. (1921). *The nature of existence*. (C. D. (Charlie D. Broad, Ed.). Cambridge, Eng. : University Press.
Mead, G. H. (1932). *The Philosophy of the Present*. Open Court.
Nelson, B. (1981, May). Remote Procedure Call. Palo Alto Research Center.
Oki, B., Pfuegl, M., Siegel, A., & Skeen, D. (1993). The Information Bus® — An Architecture for Extensible Distributed Systems. *ACM SIGOPS Operating Systems Review*, *27*(5), 58–68.
Ranadive, V. (1999). *The Power of Now*. Mc-Graw Hill.
Schulte, R. (1996). *Batch is Dead. Long Live Batch*. (Gartner Research Note No. SPA-200-159). Gartner.
Skeen, D. (1992). An Information Bus Architecture for Large-Scale, Decision-Support Environments. *Proceedings of the Winter USENIX Conference*, 183–195.
Stanford-Clark, A. (2002). Integrating monitoring and telemetry devices as part of enterprise information resources. WebSphere MQ Development, IBM Software Group.
Turner, D. A. (2004). Total Functional Programming. *Journal of Universal Computer Science*, *10*(7), 751–768.