Poster: Towards An Adaptive Event Dissemination Middleware For MMVEs

Thomas Fischer, Johannes Held, Frank Lauterwald, Richard Lenz
University of Erlangen-Nuremberg
Department of Computer Science 6 (Data Management)
Martensstrasse 3, D-91058 Erlangen
{thomas.fischer,johannes.held,frank.lauterwald,richard.lenz}@cs.fau.de

ABSTRACT

The ***M**assive **M**ultiuser **E**vent **I**nfra**S**tructure* (M^2etis) project is aimed at a generic middleware that supports the optimization of event dissemination based on a multidimensional semantic classification of event types. In this paper, we introduce the M^2etis system architecture with its core concepts.

Categories and Subject Descriptors

C.2.4 [**COMMUNICATION NETWORKS**]: Distribu Systems

General Terms

Management, Performance

Keywords

Event Semantics, Publish Subscribe, MMVE

1. INTRODUCTION

Massive Multiplayer Online Games (MMOGs) requ enormous server clusters to enable thousands of players interact within a common area of some virtual world, wh maintaining the illusion of a fluent real-time game. contrast, classical multiplayer games like Quake 3 Aren which require only one server, have shown that witho any major optimizations, the limit for a reasonable sin lation rate of 30-60 fps is reached with about 64 clients. scale beyond this limit specialized event dissemination chitectures and optimization strategies are used to expl application-specific event semantics. Based on current search and existing industrial *Massive Multiuser Virtual E vironments* (MMVEs), we assume a homogenous distribut system with identical software running on all nodes. T event types required for a particular MMOG are well-kno at design-time. Therefore, the general strategy for optimiz event dissemination may be made at compile-time as th exists a suitable patching mechanism to update all nodes the system with new content and system upgrades.

M^2etis is a generic middleware for optimized event disse ination. New MMOGs can be developed based on M^2e without the need to develop application-specific event d semination algorithms. Instead, event types are sema tically categorized using a multidimensional classification schema. The middleware can then choose readily implemented dissemination strategies that fit to the specified requirements.

http://doi.acm.org/10.1145/2002259.2002329

2. THE M^2ETIS APPROACH

Events occur between entities in the world, an entity and an avatar or between two avatars. We abstract the system

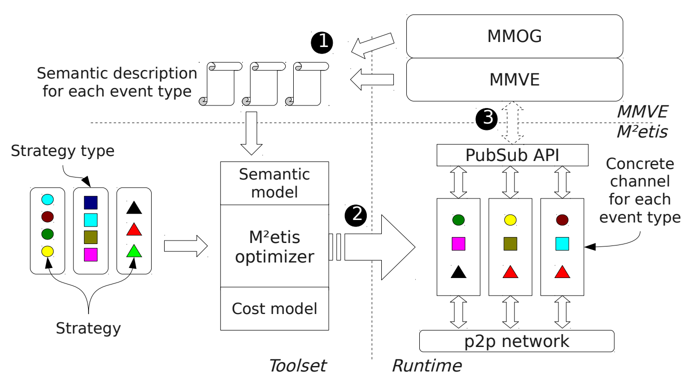

Figure 1: Architecture of M^2etis

different aspects of M^2etis. The lower left part shows the toolset of M^2etis. The toolset contains a semantic model, a cost model and the M^2etis optimizer, as well as a repository of strategy types with associated strategies. The semantic model defines the means of describing event types. In order to exploit the semantics of events, the M^2etis approach proposes a multidimensional event classification. The classification is refined from [2] to five dimensions in the context of MMVEs. The context dimension defines the set of context classes, each describing the set of nodes affected by a certain event. The Synchronization dimension describes the degree of order a certain channel provides for its context. Persistency means, each event of an persistent channel has an effect on the state of the application. Security is currently not considered in this research and therefore not described. In contrast to [2], we redefine validity as a mapping, defining all messages relayed to the application. Each dimension has a certain number of classes each defining a different semantical behavior along the dimension. An event type has exactly one class along each dimension. The cost model defines the actual rules and heuristics to rate each fitting strategy for the optimizer, taking all semantic requirements and non functional parameters into account.

M^{2}etis provides different implementations for each strategy type,which defines the interface of event processing for each implementation, named strategy. Strategy types are therefore sets of strategies, implementing the same functionality, only differing in their non-functional properties. Strategy types and their association with the dimensions is the step from the semantic description to the actual publish/subscribe channel. To generate a channel, an optimizer configures a channel with different strategies, one of each strategy type. The optimizer's task is to estimate the costs for each available strategy and select the best one for a certain scenario, while making sure that only strategies with the appropriate semantic characteristics are chosen. Each strategy is annotated with cost information that is used together to choose the best strategy of each type in the optimization step.

The lower right part of Figure 1 shows how M^{2}etis is seen at runtime. M^{2}etis provides a network layer for the MMVE, hiding the used *Peer-to-peer* (P2P) network with a channel-based publish/subscribe system. Every channel corresponds to an event type and is highly optimized for that event type.

The workflow for application development using M^{2}etis requires that the MMVEs' engineer identifies all event types which need to be distributed among the nodes of the MMVE and describes each according to the semantic model in step 1 of Figure 1. These semantic annotations are processed via the M^{2}etis optimizer according to the semantic model and the cost model. The optimizer assembles a communication channel for each event type using the best matching strategy for every strategy type leading to an optimized distribution for every event type (step 2). At runtime, the MMVE uses the provided publish/subscribe API to publish and receive its events (step 3) without any further knowledge of the optimizations.

3. PROOF OF CONCEPT

As a proof of applicability of the M^{2}etis approach, we implemented a prototypic game called *tri6*. This game is a fast-paced racing game in which each player is able to perform different actions. As discussed in the context of the architecture and usage of M^{2}etis, the framework is used as the networking component of this game.

The M^{2}etis prototype itself currently consists of the prototypic implementation of the publish/subscribe component. We chose C++ and *key-based routing* (KBR)-overlays as possible underlying networks. In order to minimize the overhead introduced by the flexibility of such a framework e.g. regarding message size or stack depth caused by function calls, we use *template metaprogramming* (TMP) and *policy based*-design to create the optimized channels at compile-time. The different optimization strategies for each dimension are implemented as policies. With TMP, it is possible to derive custom-tailored message headers, depending on the chosen strategies. This ensures small message sizes with a high payload ratio. All described design decisions ensure a small footprint at runtime.

4. DISCUSSION

M^{2}etis addresses a gap in existing publish/subscribe architectures as there exist no middleware architectures using semantic classification of event types to optimize their dissemination. Existing solutions like [1, 3] use a uniform treatment of all event types. By the support of different MMVE-specific and generic strategies and the extensible architecture M^{2}etis tries to close this gap. One constraint of the proposed approach is the focus on simple event processing without the possibility to process complex events. As the performance requirements in the domain of MMVE are very demanding the question is whether it is desirable to process complex events. In our studies, we did not encounter any application requirements which made it absolutely necessary. The publish/subscribe paradigm also states a controversial interaction scheme, as some MMVE use a request/reply paradigm. Which paradigm is the best-fitting model, depends on the rate of subscriptions and unsubscriptions, as these form the overhead of the communication. Another point of discussion is the P2P based approach of M^{2}etis. In times of cloud computing one may also argue, that MMVE clusters may be extended by the usage of cloud resources. This argumentation is valid and in our opinion a field, which should be analyzed in the context of MMVEs, but our approach is theoretically applicable in cloud environments, especially enabled by its flexibility gained by the underlying network abstraction. The idea of the exploitation of the different event semantics on the other hand promises to be an approach to optimized event dissemination architectures not only fitting for MMVEs.

Based on the existing prototype, further work must be done. Currently we are adapting different existing optimization algorithms. Based on this variety of algorithms, measurements will be taken by suitable simulation. As a foundation for the planned evaluation and performance measurements, a simulator is currently under development. Another part of our ongoing research is the extension of the M^{2}etis system model to support stateful channels and persistence. We understand the current prototype as the first step towards an adaptive middleware for MMVEs.

5. REFERENCES

[1] A. Bharambe, J. R. Douceur, J. R. Lorch, T. Moscibroda, J. Pang, S. Seshan, and X. Zhuang. Donnybrook: Enabling Large-Scale, High-Speed, Peer-to-Peer Games. *SIGCOMM Comput. Commun. Rev. (CCR)*, 38(4):389–400, 2008.

[2] T. Fischer, M. Daum, F. Irmert, N. C. P., and R. Lenz. Exploitation of Event-Semantics for Distributed Publish/Subscribe Systems in Massively Multiuser Virtual Environments. In *IDEAS '10*, Montreal, QC, Canada, 2010.

[3] S.-Y. Hu, J.-F. Chen, and T.-H. Chen. VON: A Scalable Peer-to-Peer Network for Virtual Environments. *IEEE Network*, 20(4):22–31, July 2006.