Chapter 1

Introduction

O Spouse of Zeus, Lady of heavy anger,
thou wert not to be for long
without tidings thereof:
so swift a messenger hastened to thee.

Callimachus, Hymn IV to Delos

The notion of a messenger as means to efficiently deliver and announce valuable information is an old and well-known concept, from Hermes and Iris, the messengers of the Olympian Gods, over Pheidippides, the first marathon runner, to the early bike messengers in Chinese Qing Dynasty. In the Internet age, an event notification system delivers information about selected events at providers' sites to its clients. Similarly to a messenger, an event notification service notifies about events, which are occurrences of interest to the service's clients. Figure 1.1 illustrates providers and clients of event notification services: Providers of events are sensors, clocks, and application programs. Examples of events include the reading of a sensor at different times, traffic jam information, and the publishing of a book. Users, or clients, of such a system are human beings or application programs. The clients receive notifications, for example via hand-held devices, Internet pages, or email.

In recent years, the communication paradigm of event notification has developed from a communication model on operating system level to a key concept of information delivery, specifically for Internet applications. In the 90's, a set of Internet-based services, such as CDF channels [Ell97], claiming to push content to interested clients caused a *push-hype* [KW97]. Currently, several solitary services exist for various application fields informing their clients, for example, about new publications of a publishing house [Spr01] or changed webpages [Ari99]. According to Olken [OJM98a], it "is the ultimate goal to have a well understood set of high level event services available to any distributed control application".

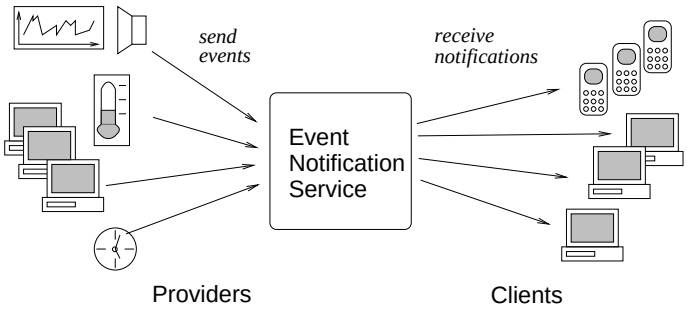

Figure 1.1: Observation and notification of events

At present, new applications emerge that call for a service that integrates event-information from different sources. One example is a traveller information system that uses data about public transport, highway traffic news, and weather information. Here, a notification service could actively inform its clients about traffic problems or suggest a good travelling route with panoramic view depending on the traffic and weather condition. Other examples for new applications are remote facility management for the monitoring of large distributed areas and extensive logistics control for warehouse management and delivery companies.

Integration requires the notion of composite events that are temporal combinations of simpler events, e.g., a bus leaving *after* the train arrives. Additionally, the event sources might handle events differently. For example, the weather source regularly reads a temperature sensor, while the traffic news source reports only changes in the road status, e.g., from free to congested road. Furthermore, the event notification service may switch between event sources: New sources become available and others cannot be reached temporarily. However, the service's clients should not be forced to redefine their interest in order to cover these changing event sources. Current service implementations either do not support integration or they are too rigid and do not provide sufficient solutions. Moreover, the performance of such a service is subject to strong demands: A growing number of event providers and system clients, and the coverage of sources with high event frequencies, such as location-sensors for vehicles, result in high system traffic.

The missing link for empowering existing applications and enabling novel solutions is a service that flexibly supports various and changing application requirements and enables efficient client-driven integration of information from various sources. Note that in our context *integration* does not represent the opposite concept of 'distribution' but a logical integration similar to the integrative data handling in distributed databases. Integrating event notification services may be implemented as distributed services. This thesis presents an integrating event notification service that adapts to different application requirements in a number of ways: (1) The handling of composite events can be adapted to the needs of the service applications and clients. (2) The service flexibly reacts to application-dependent error sources that cause, e.g., message delays. (3) In order to gain maximum performance, the filter process for primitive events adapts to changing value distributions in events and client interests. (4) Several methods for the filtering of composite events are supported, their selection depends on the profile distribution.

In this first chapter, we introduce the paradigm of event notification as a communication model and examine the potential of an adaptive integrating event notification service. After outlining the challenges

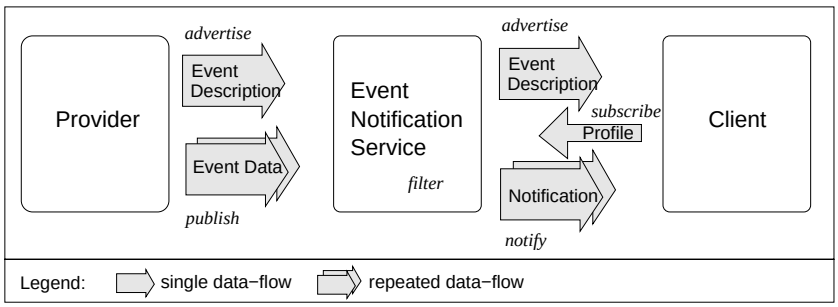

Figure 1.2: Data-flow in an event notification service

involved in designing such a service, we explain how the thesis addresses these issues. The contributions of this work are summarized briefly. Finally, the structure of the remainder of this document is given.

1.1 Event Notification Services

In this section, we introduce the general structure and characteristics of event notification services (ENS[1]). We direct the focus of the thesis to application level services and give a brief overview of research concerned with ENS on that level.

Fundamentals of Event Notification Services. An event notification service connects providers of information and interested clients. The service informs the clients about events on the provider's site. Events can be, for instance, changes of existing objects or the creation of new objects such as the publication of a new web-page. The service learns about events by means of event messages reporting the events. The event messages are compiled either by the providers (push) or by the service actively observing the providers (pull). Clients define their interest in certain events by means of personal profiles. The profile creation is based on the profile definition language of the service. The information about observed events is filtered according to these profiles and notifications are sent to the interested clients.

Figure 1.2 depicts the data-flow in a high-level architecture of an ENS. Keep in mind that the data-flow is independent of the delivery mode, such as push or pull. The tasks of an event notification service can be subdivided into four steps. First, the observable event (message) types are to be determined and advertised to the clients. This advertisement can be implicit, for example, provided by the client interface or due to information the client has about the service. Second, the clients' profiles have to be defined via a client interface, they are stored within the service. Third, the occurring events have to be observed and filtered by the event notification service. Before creating notifications, the event information are combined in order to detect composite events (e.g., a bus leaving after the train arrives). The messages can be buffered to enable more efficient notification (e.g., by merging several messages into one notification). Fourth, the clients have to be notified, immediately or according to a given schedule.

Note that the expressiveness of the profile definition language and the filter capability of the service

[1]We use the acronym ENS for both, event notification services and event notification systems. Additionally, the plural form is also ENS.

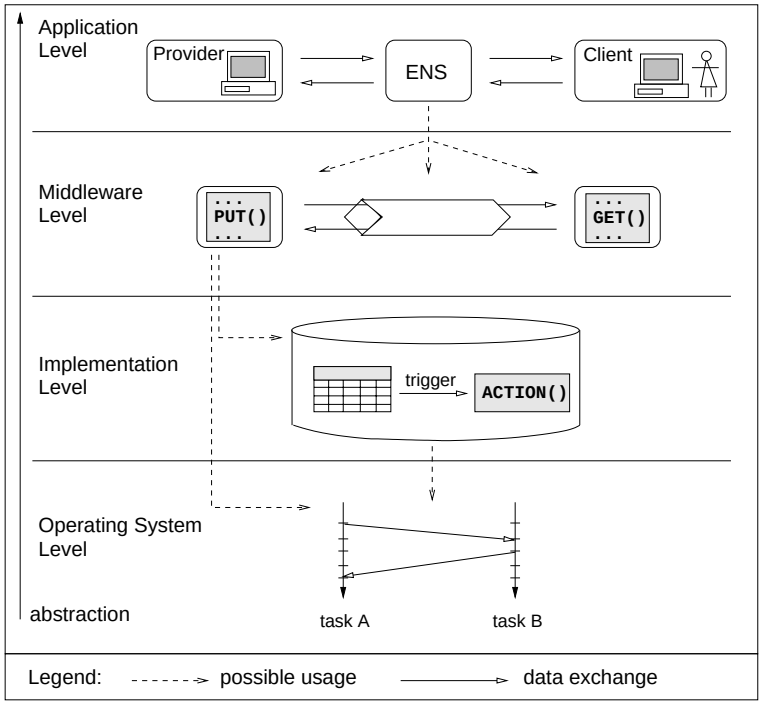

Figure 1.3: Levels of event-based communication

are interdependent. Event notification services typically support either subject-based or content-based subscriptions (profiles). Subject-based services distribute event messages according to their predefined topic or subject, e.g., send all travel information. Content-based services filter events based on the content of event messages, e.g., notify about delays of the 10 o'clock train from Berlin to Paris. This thesis focuses on event notification services supporting content-based filtering, since it allows for fine grained client profiles that are necessary in the addressed integrating applications.

Levels of Event-based Communication. As already pointed out, the event/message paradigm is a well known communication model. ENS can be found at different system levels of abstraction, starting with the application level down to the operating system level (see Figure 1.3). Note that event-based communication on one level does not necessarily require an event concept on the next-lower level.

Application-level services have been well studied since the beginning of ENS in 1968 [Sal68]. Examples for applications that benefit from event notification services are digital libraries, stock tickers, and traveller information services. The services handle application-dependent client profiles that are addressed to the system as a whole. The system observes and filters events from external sources. Currently, several independent implementations of event notification services already exist, such as SIFT [YGM95], Siena [CRW01], OpenCQ [LPT99a], Gryphon [SBC98], and LeSubscribe [PFJ01].

On the middleware level, distributed software components can communicate, e.g., by use of system-specific events that may encapsulate and transport events from the higher application level. Events from the higher application level are handled as mere data to be processed – their inner structure and interpretation is not known at this level. Examples for middleware-services are BEA Message Service [BEA00],

MQ Series [IBM95], and the specifications for CORBA notification service [OMG99] and Java Message Service [Sun99].

Services on the implementation level notify about internal events, e.g., inserts into a database table. Other examples are event notification in distributed programming [SCT95], debugging [OCH91], and monitoring of distributed systems [MS97]. Middleware-level events can be mapped onto these events. Requests on the implementation level are then translated to operating system calls. Event-based communication examples are triggers and rules in active database systems (aDBS).

On the operating system level, events are used for synchronization of system tasks [TG99] and transaction monitors [BN97]. This level also encloses low-level event handling in programming languages and GUI frameworks, such as in Java Swing [WC99] and SWT [Obj01].

In this work, we concentrate on application-level services; we consider services on middleware and implementation levels to be base-techniques for the implementation. We do not consider event-driven communication on the operating system level.

Event Notification Services: Past, Present, and Future. One of the earliest forms of electronic ENS came from work on Selective Dissemination of Information (SDI). SDI was designed as an automatic way of keeping scientists informed of new documents published in their areas of specialization. A scientist could create and modify a client profile of keywords that described his or her interests. Then, an SDI service, such as SIFT [Sal68], used the profile to match the keywords against new articles to find the documents most relevant to the scientist's interests. While SDI was implemented on a large scale in the 70's, it was used far less than predicted due to difficulties of profile construction [PS79].

In the 80's and beginning 90's, researchers concentrated on the development of fine-grained and effective filtering methods. Information filtering systems were introduced as SDI systems for unstructured or semi-structured data. Belkin and Croft [BC92] showed that information filtering and information retrieval (IR) are almost identical at an abstract level. IR uses queries as specifications of information needs, and is concerned with single uses of the system, whereas information filtering employs client profiles to satisfy long-term goals of the clients. Studies on client models and profile generation aimed at improving the clients' acceptance of filter systems.

With the expanding use of the Internet in the 90's, push-technology was celebrated as the solution for information delivery. A great variety of event notification services was developed for every day use, filtering not only documents (as in SDI) but notifying its clients about various events, such as changed shareholder values, newspaper articles, and sports news. Ninety percent of these services were already off the market two years later. The services required special software to be installed at provider and client sites, supported only coarse-grained profiles, and did not cooperate with other products.

Today, research on wide-area event notification services focuses on efficient and scalable filtering in a distributed environment [ASS$^+$99, Car98, FLPS00, PFLS00]. Available event notification systems are either implementations for specific applications (e.g., Springer Link Alert [Spr01] and Hermes [Her03] for Digital Libraries) or academic research prototypes that address very particular features, such as an expressive profile/filter language (e.g., JCQ [LPT99b]) or performance (e.g., Elvin [FMK99]).

Independent solitary services are no longer appropriate for newly emerging applications that need

notification facilities integrating event information from heterogeneous sources. The quantity of available event information, their temporal character, and possible interrelations may burden the clients with unnecessary load. For example, traveller information systems use data about public transport, highway traffic news, weather and tourist information. Therefore, our event notification service supports these applications by integrating information that is provided by different sources using different event types and semantics — its handling of composite events adapts to different sources, event observation methods, and application-specific client preferences.

1.2 Challenges and Solutions Envisioned

The goal of this thesis is the design and evaluation of an adaptive integrating event notification service. This section outlines the challenges in the design and names the solutions at which this thesis aims. We focus on two main challenges for the design: (1) the detection of composite events provided by various sources and (2) the performance of the system under changing system load.

(1) Composite events are essential for an integrating service; their observation and composition has to reflect the character and semantics of the event sources and applications. The service has to provide integration of both event data and event sources. Event composition has to take into account not only the client profiles but also source characteristics and application requirements. At the event data level, various event semantics, e.g., due to various event observation methods at the sources, have to be considered. At the service level, various event information qualities due to the respective event source characteristics have to be taken into account. Therefore, the event notification service needs to adapt to the different structures and qualities of the sources in order to combine the event information. Our solution is the definition of an event semantics that controls event composition by adaptable parameters. Integration of events from different sources requires the consideration of various error-sources, such as incorrect timestamps and observation delays. We propose both a strategy for the handling of events' timestamps and a time system extension. Both approaches help to minimize the temporal delays and to increase the event detection accuracy.

(2) Performance is important due to the nature of ENS: Events occur at a certain frequency, e.g., up to thousands per second in remote monitoring applications. A large number of clients has to be served in time. This challenge may be addressed at two levels: local performance of event processing and scalability via distribution of the service. In this thesis, we focus on the first approach. Our work makes use of the findings of the extensive research in the field of ENS distribution.

The performance requirements of the system are addressed in two approaches: for the filtering of primitive events as well as for the filtering of composite events. We propose a filter algorithm for primitive events that takes the distributions of events and profiles into account. For the filtering of composite events, we propose a filter algorithm that integrates the detection of composite events into the detection of primitive events. Both algorithms significantly improve the performance for typical event and profile distributions in event notification services.

1.3 Contributions of this Thesis

The central contribution of this thesis is the design and evaluation of an adaptive integrating event notification service. This work is based on a thorough evaluation of selected application scenarios and their requirements for event notification services. Equipped with the results of an analysis of existing event notification systems, we propose a novel service design: The service is controlled by a set of system parameters that allow for automatic adaptation of service behavior. In particular, two aspects have been addressed: the handling of composite events and the efficient filtering of events. The proof of concept is given by the implementation of the ENS prototype A-MEDIAS and the application of our system to a facility management scenario.

In detail, this work contributes to research in event notification services as follows:

Analysis of Application Scenarios. We study relevant applications for their specific demands regarding event notification services. We consider selected scenarios for new applications in remote monitoring and control, logistics control and traffic support, and in web-based education. Abstracting from the scenarios, we classify the requirements for event notification systems as follows: elaborated event model including composite events, support for various event observation strategies, flexible event composition, cooperation and integration of event sources, and scalability and performance. These requirements are elaborated in detail and form the starting point for the review and analysis of existing event notification systems.

Review and Analysis of Event Notification Systems. As a basis for the system analysis, we define an abstract model for event notification systems consisting of reference model, event model, and event processing model. The model serves as support for our analysis of existing event notification services. Our model and a first analysis of systems are published in [HF99a]. We present a substantial review, discussion, and comparison of the relevant systems and related technologies. Special focus is given to their support for the requirements. Our analysis shows that current service implementations do not provide a sufficient solution. We can distinguish two approaches: The first type of event notification services does not support composite events at all – event integration is not possible with these services. The second type does support event composition, but the events are processed in a fixed, predefined manner. Flexible adaptation to differing or changing sources is not possible. Selected results of the analysis have been presented in [FHS98, SHF00]. This thesis includes a more extensive analysis considering the latest developments.

Design of an Adaptive Integrating Event Notification Service. The central contribution of this thesis is the design of a service that is highly customizable to different applications, and thereby supports the integration of various event sources. The two main-issues of (1) the detection of composite events and (2) the performance of the system are addressed in four steps:

- **Adaptable Event Composition.** As shown in the requirements analysis, ENS should adapt their event composition strategy according to the applications' needs. Existing event notification services lack flexibility in the handling of composite events: Each semantical variation of event

compositions has a separate operator. Addressing this weakness, we apply and extend event ordering concepts from active databases in our context of event notification systems. We introduce an event algebra that mirrors the semantical facets of composite events in different application fields. In a unique way, the semantics of composite events defined by this algebra are adaptable to changing application and client demands. We do not introduce a variety of new operators but employ generic operators that are parameterized to cover the semantical variations. The algebra has been published in [HV02]. The algebra sets the basis for both a sophisticated profile definition language and a flexible filter component.

- **Accurate Event Composition.** The second step concentrates on error sources for event composition, such as observation and timestamp delays. We analyze the implications of composite events that are integrated from different sources on the event filtering. Our analysis shows the influence of observation and timestamping methods on the quality of composite-event filtering. Several strategies for error-correction are proposed to avoid incorrect filter results. We introduce a measure for temporal quality of event information and its application within event notification services. Selected results of our analysis have been published in [Hin01].

- **Distribution-dependent Filtering Algorithm for Primitive Events.** Our investigations of the influence of system and network characteristics on the service's performance substantiates the importance of efficient event filtering. We develop an efficient filtering algorithm for primitive events that utilizes the application-dependent distributions of events and profiles. The algorithm is based on selectivity measures for events and profiles. Extensive simulations and tests have shown the applicability and usefulness of the selectivity-measures. For certain cases, filtering time was reduced by about 90 percent [HB02].

- **Efficient Algorithm for Filtering of Composite Events.** For the evaluation of composite events, we propose a new filter algorithm. In contrast to existing filtering approaches that have a separate composition-phase after the primitive-event filtering, our algorithm uses an integrated single-pass approach. The second phase is omitted and the filtering of composite events is integrated into the simpler handling of primitive events. The filter time of each composite event is reduced and the overall filter increases. The filtering of composite events is accelerated by about 85 percent [Hin03b].

Proof of Concept. As a proof of concept, we implemented our design of an integrating event notification service introduced in this thesis as the prototypical event notification system A-MEDIAS. Our formal specification of the algebra for composite events is translated into a language for profile definition. The filter algorithms for primitive and composite events have been implemented and tested within our prototype. We show the usage of A-MEDIAS in the context of a facility management scenario. The adaptive approach of our A-MEDIAS system has also been presented in [Hin03a]. Having designed our adaptive integrating event notification service, an evaluation of the service along the requirements summarizes the strengths of the design.

1.4 Thesis Outline

This thesis is organized in three parts. The first part (chapters 2 to 4) provides the basis for our work. We analyze application scenarios for their requirements to ENS, present a model for event notification services, and evaluate existing approaches.

Chapter 2: Analysis of Application Scenarios We present three application domains, where the employment of an adaptive integrating event notification service is required. Given the application scenarios and the observations made, we identify six basic requirements for an adptive integrating event notification service.

Chapter 3: A Model of Event Notification Services We introduce an abstract model for event notification services consisting of a reference model, an event model, and an event processing model. The model serves as a basis for the analysis of related event-based approaches.

Chapter 4: Review and Analysis of Event Notification Services In this chapter, we evaluate the state of the art in event-driven services with regard to the identified requirements. We analyze related services and technologies that are directly relevant to or have influenced this thesis.

Our adaptive integrating event notification service is developed as an answer to the analysis that reveals the lack of flexibility in existing approaches. In the second part (chapters 5 to 8), we present our design of this service.

Chapter 5: Adaptable Event Composition We introduce an event algebra that defines the semantics for composite events. The algebra is first introduced informally. The formal definition gives an insight into the details of temporal event interaction.

Chapter 6: Accurate Event Composition We analyze possible error sources that influence the quality of the event composition. Special attention is given to observation and timestamping methods. We propose the notion of temporal accuracy as a measure for temporal quality of event information and introduce error-handling methods.

Chapter 7: Efficient Event Filtering The results of an ENS performance simulation substantiate the importance of efficient event filtering. We propose an adaptive filtering algorithm for primitive events. It utilizes the application-driven distributions of events and profiles as a selectivity measure. For the filtering of composite events, we propose a new filter algorithm that reduces the evaluation of composite events to the simpler filtering of primitive events.

Chapter 8: Implementing Adaptivity We discuss how to realize adaptivity in an integrating event notification service. We propose the translation of client queries according to application profile and source profiles. Additionally, strategies to determine the distributions of events and profiles and to adapt the filter algorithms are discussed.

The third and final part (chapters 9 to 11) serves as a proof of concept: We introduce the implementation of our prototype and present selected results of efficiency tests. The conclusion summarizes the research presented in this thesis.

Chapter 9: The A-MEDIAS *System* We present our A-MEDIAS system – a prototypical implementation of an adaptive integrating ENS that realizes the theoretical design concepts introduced throughout this thesis. We give an overview of the architecture of the system and introduce the system's profile language. Special focus is given to the filter implementation.

Chapter 10: Experimental Results To illustrate the system's advanced features, we show the application of our prototype in case studies from a facility management scenario. The results of extensive performance tests show the efficiency of the filtering and the additional performance gains for adaptive system behavior.

Chapter 11: Conclusion We evaluates the A-MEDIAS system along the requirements and give an overview of the contributions of this thesis. We review insights gained in the course of this investigation, point out open issues, and suggest directions for future research.