SFB 627 Bericht Nr. 2006/08

Performance Aspects of Large-scale Location-based Services

Datum:	11. Juni 2006
Autor(en):	Michael Scharf, Bruno Arbter, Frank Dürr, Thomas Schwarz, Oliver Simoneit
CR Klassifikation:	C.2, C.4, I.6, H.4
(c) 2006	Michael Scharf, Bruno Arbter, Frank Dürr, Thomas Schwarz, Oliver Simoneit

Universität Stuttgart

INSTITUT FÜR KOMMUNIKATIONSNETZE UND RECHNERSYSTEME
Prof. Dr.-Ing. Dr. h. c. mult. P. J. Kühn

Projektbericht

Auftraggeber: SFB Nexus

Titel: **Performance Aspects of Large-scale Location-based Services**

Verfasser: Michael Scharf (Editor)
Bruno Arbter
Frank Dürr
Thomas Schwarz
Oliver Simoneit

Version: SFB 627 Bericht Nr. 2006/08

Datum: June 2006

Kurzfassung: The Nexus vision is a flexible platform providing spatial world models for context-aware services. Realizing such a service platform in a scalable way is a challenging issue. Therefore, the performance of components of the Nexus system is addressed by different sub-projects within the center of excellence. This document is the final report of a taskforce that has coordinated these activities in an interdisciplinary way. The report first gives an overview of different methods used to evaluate the performance and scalability of the Nexus system. Second, several studies on aspects of particular importance are presented. The report concludes with an analysis of different usage scenarios for large-scale location based services with respect to performance and scalability issues.

CR classification: C.2, C.4, I.6, H.4

1 Introduction

The Nexus vision is a flexible platform providing spatial world models for context-aware services. Realizing such a service platform in a scalable way is a challenging issue. The performance of different components is addressed by a number of different projects in the center of excellence, which have been coordinated in the so-called “Taskforce Performance”. In particular, the evaluation of the Nexus system architecture under leadership of project A1 has been coordinated within this taskforce. This document is the final report of this taskforce and covers three different aspects, which have been worked on in the taskforce:

- An overview of methods that can be used to evaluate the performance and scalability of the Nexus system, as well as relevant metrics.
- Detailed studies on several aspects of particular importance, in particular performed by project A1.
- Reference scenarios for the usage of large-scale location-based services that can act as starting point for evaluations of components of the Nexus system.

The work of the taskforce is closely related to other working groups within the Nexus project, in particular

- “Arbeitsgemeinschaft Mobilitätsmodellierung” (mobility modeling working group),
- “Taskforce Mobile Objects”, and
- “Taskforce Betreiber” (service operator taskforce).

Issues related to mobility modeling and economic aspects are covered in the reports of these working groups and are therefore not considered in detail in this document.

2 Methodology

2.1 Overview

This section provides an overview of methods to evaluate the performance of the Nexus platt-form.

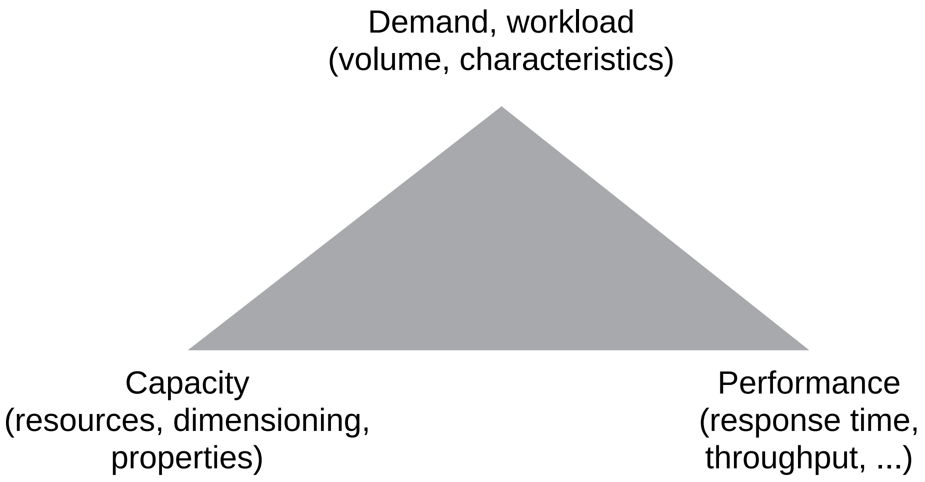

Demand, workload
(volume, characteristics)
Capacity
(resources, dimensioning, properties)
Performance
(response time, throughput, ...)

Figure 2.1: Three aspects of performance evaluation

Figure 2.1 illustrates that performance evaluation has to deal with three different aspects: The workload, the capacity and the performance. The design of a system architecture or the performance evaluation of a system requires usually two of these three aspects to be known:

1. If the system capacity and the workload is known, critical system parameters such as response times, throughput, etc. can be obtained by modeling and performance evaluation, e. g., by analytical or simulation methods.

2. If the capacity is known and there are given performance requirements (Quality of Service requirements, see below), one can determine the maximal possible load under which these constraints can be met.

3. If both the workload and performance requirements are known, guidelines for the design of the system can be developed, such as the required number of resources and their dimensioning, etc.

In Nexus, in principle all three different approaches are possible, depending on what is known of the characteristics and the components. In the following, the important aspects in particular of the first approach are surveyed in more detail.

2.2 Performance Metrics

2.2.1 Network-level Quality of Service (QoS)

Quality of Service (QoS) is an important requirement of users concerning the performance of technical systems, in particular, of communication networks. According to [2], this term is defined as follows:

> “The collective effect of service performance which determines the degree of satisfaction of a user of the service. [...] The quality of service is characterized by the combined aspects of service support performance, service operability performance, serveability performance, service security performance and other factors specific to each service.”

For communication networks, *network performance* is of particular importance:

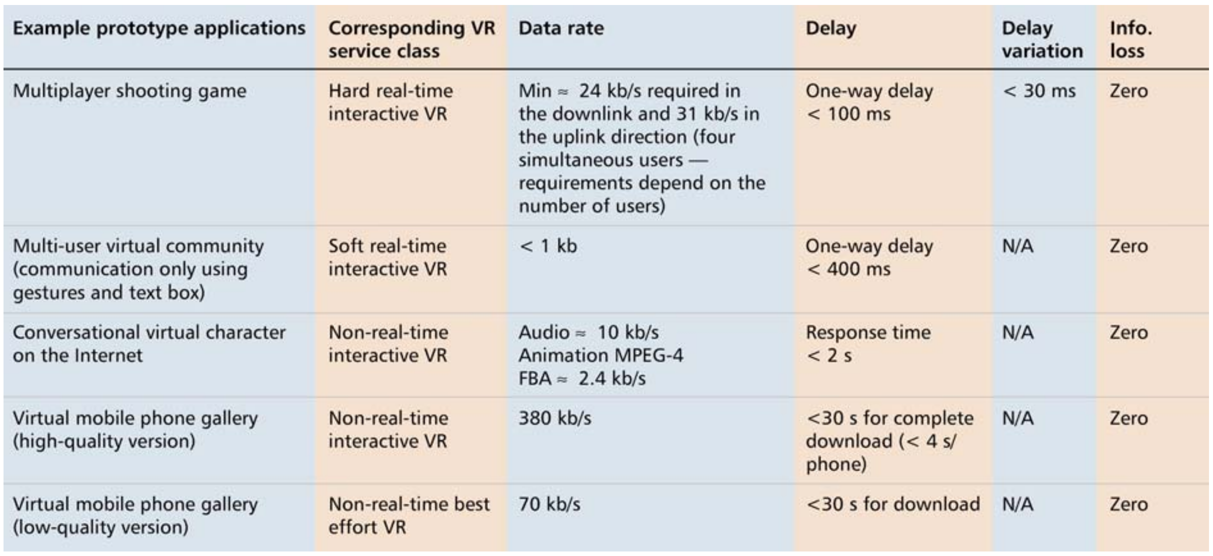
> “The ability of a network or network portion to provide the functions related to communications between users. [...] Network performance applies to the Network Provider’s planning, development, operations and maintenance and is the detailed technical part of QOS, excluding service support performance and human factors. [...] Network performance measures are meaningful to network providers and are quantifiable at the part of the network to which they apply. Quality of service measures are only quantifiable at a service access point. [...] It is up to the Network Provider to combine the Network Performance parameters in such a way that the economic requirements of the Network Provider, as well as the satisfaction of the User, are both fulfilled.”

Public mobile networks (GSM, UMTS, etc.) can support data transport with different quality of service classes:

Table 2.1: Quality of service classes in UMTS (see e. g. [3])

Traffic class	**Typical application**	**Latency**		**Small error rate**	**Guaranteed data rate**
		Sum	**Jitter**		
Conversational	VoIP, video/audio conferencing	small < 100 ms	small	no	yes
Streaming	Broadcast services (audio, video), video playback	small < 250 ms	small	no	yes
Interactive	Web browsing, interactive chat, games, M-commerce	medium	not critical	yes	no
Background	E-Mail, SMS, data base retrieval	not critical	not critical	yes	no

The quality of service requirements depend on the type of application. In the Nexus project, different types of applications are developed, ranging from streaming applications (e. g., 3D visualization of world models) over interactive applications (e. g., access to information in the world model) to background traffic (e. g., synchronization of databases on different context servers). It should be noted that many Nexus applications do not have very strict real-time requirements, i. e., the response time jitter is rather uncritical. An exception may be virtual reality applications, which are particularly challenging, since they both require high data rates, low delay, and sometimes even low delay jitter (see Figure 2.2).



Example prototype applications	Corresponding VR service class	Data rate	Delay	Delay variation	Info. loss
Multiplayer shooting game	Hard real-time interactive VR	Min ≈ 24 kb/s required in the downlink and 31 kb/s in the uplink direction (four simultaneous users — requirements depend on the number of users)	One-way delay < 100 ms	< 30 ms	Zero
Multi-user virtual community (communication only using gestures and text box)	Soft real-time interactive VR	< 1 kb	One-way delay < 400 ms	N/A	Zero
Conversational virtual character on the Internet	Non-real-time interactive VR	Audio ≈ 10 kb/s Animation MPEG-4 FBA ≈ 2.4 kb/s	Response time < 2 s	N/A	Zero
Virtual mobile phone gallery (high-quality version)	Non-real-time interactive VR	380 kb/s	<30 s for complete download (< 4 s/ phone)	N/A	Zero
Virtual mobile phone gallery (low-quality version)	Non-real-time best effort VR	70 kb/s	<30 s for download	N/A	Zero

Figure 2.2: Quality of service requirements of Virtual Reality applications (from [4])

2.2.2 Application-level Quality of Service

Databases storing world model information are a central component in the Nexus system. For databases, the most important performance metric is the *response time*. Another important metric is the *throughput* of the system, e. g., given by the number of transactions that can be handled in a certain time period. As the Nexus architecture is a distributed system, the overall performance (response time, throughput) depends on the performance of the individual components. For such Web Service based systems, [1] lists further performance metrics:

- *Response time*
- *Throughput*
- *Availability*
- *Cost*

According to [1], the delays encountered by a request may be decomposed into *service times*, i. e., the time spent using various resources such as processors, disks, and networks, and *waiting times*, which are spent waiting to use resources that are being held by other requests. In typical Web Service systems, up to fifty percent of the response time may result from the reaction time of the database server.

This makes it important so meet certain service levels, such as a minimum throughput, a minimum availability, and a certain distribution of response times. [1] mentions that typically 60% of search requests are aborted by the user if the response time is 4-6 s, and up to 95% if it exceeds 6 s. As a consequence, both for response time and throughput, mean values as well as quantiles (e. g., 95%-quantile) have to be considered.

2.2.3 Quality of Context

The system performance is not the only important characteristic in the Nexus system. Closely related are metrics describing the quality of information that are provided by the system. This quality can be parametrized by *Quality of Context* metrics. In [5], this term is defined as follows:

> Quality of Context (QoC) is any information that describes the quality of information that is used as context information.

According to [5], the most important QoC-parameters are the following:

1. *Precision*: How exactly the provided context information mirrors the reality.
2. *Probability of Correctness*: Probability that a piece of context information is correct [in face of sensor failures]
3. *Trust-worthiness*: How likely it is that the provided information is correct
4. *Resolution*: Granularity of information
5. *Up-to-dateness*: Age of context information

In this report, *precision* mainly describes the precision of location information, *probability of correctness* quantifies whether the information in the world model is correct/consistent, *trust-worthiness* characterizes whether one must trust the context service provider, *resolution* the level of detail of the models, and *up-to-dateness* the maximum age of information in the world model. It should be noted that these metrics typically can be characterized by values of a certain granularity. Examples:

- Precision: very low (>100m), low (10m-100m), medium (1m-10m), high (10cm-1m), very high (1cm-10cm)
- Up-to-dateness: very low (>1h), low (10min-1h), medium (1min-10min), high (10s-1min), very high (1s-10s)

As shown in the reference scenarios at the end of this report, for most Nexus applications a lower bound of these metrics can be given, which is the minimum value required for a certain service.

Quality of Service and Quality of Context are closely related issues. As shown in [6], and as also discussed in later sections of this document, there is a trade-off between QoS and QoC. This can be easily seen from the fact that obtaining world model information with good quality (high QoC) may require much more processing and thus implies a longer response time (bad QoS).

2.2.4 Scalability Metrics

Many distributed systems must be scalable, which means that they must be economically deployable in a wide range of sizes and configurations. However, there are different notions and definitions of the term *scalability*:

- “The ability to grow the power or capacity of a system by adding components.” [7]
- “[...] scalability is defined as the ability to support large numbers of accesses and resources while still providing adequate performance.” [8]
- “Scalability means not just the ability to operate, but to operate efficiently and with adequate quality of service, over the given range of configurations.” [9]

- "[Scalable systems] must be capable of being deployed efficiently at both large and small scales." [10]
- "The concept [of scalability] connotes the ability of a system to accommodate an increasing number of elements or objects, to process growing volumes of work gracefully, and/or to be susceptible to enlargement". "When we say that a system is unscalable, we usually mean that the additional cost of coping with a given increase of traffic or size is excessive, or that it cannot cope at this increased level at all." [11]

Furthermore, [11] distinguishes between different aspects of scalability, in particular *load scalability* and *structural scalability*:

- "Load scalability is the ability of a system to perform gracefully as the offered load traffic increases". Factors that can undermine load scalability are e. g. the scheduling of shared resource, the scheduling of a class of resources in a manner that increases its own usage (self-expansion), and inadequate exploitation of parallelism.
- "Structural scalability is the ability of a system to expand in a chosen dimension without major modifications of its architecture". This means that its implementation or standards do not impede the growth of the number of objects it encompasses.

In addition to this, [11] also defines the term *space scalability* for the case that memory requirements do not grow to intolerable levels as the number of items it supports increases, and the term *space-time scalability* for systems that continue to function gracefully as the number of objects it encompasses increases by orders of magnitude. However, these different aspects of scalability are not completely orthogonal.

A variety of scalability metrics have been developed for massively parallel computation in order to evaluate the effectiveness of a given algorithms. Three kinds of metrics have been reported: speedup metrics, efficiency metrics, and scalability metrics [9]. A typical metric in the multiprocessor domain is the *fixed size speedup*, also called "Amdahl's law", which is defined as the ratio of sequential execution time to parallel execution time. Another possibility is the *fixed time speedup* or "Gustafson's law", which is the sum of the work done by all parallel processors divided by the work done by a sequential system in the same time (see [10]). References to further scalability metrics are given in [9]. However, these concepts cannot be extended directly to distributed systems because of their higher complexity and their heterogeneity.

The objective of scaling up a distributed system is either to support more throughput for a fixed response time, to reduce the response time for a given throughput, or a combination of these. This motivates a *power* metric (cf. reference in [10])

$$P = \frac{\text{throughput}}{\text{response time}} = \frac{\lambda}{T} .$$

For any delay function $T(\lambda)$, the maximum power point is the place where the tangent from the origin touches the $T(\lambda)$ curve. [10] defines a metric to evaluate the scalability of a distributed system based on performance and cost. The scalability metric relates the power of a system to the cost of obtaining that power: "If the power per invested dollar can be improved (or maintained constant) by evolving the system configuration, the distributed system is said to be scalable from its old state to the new state”. The *scalability function* from a state S to a new state S' is

$$\Psi(S, S') = \left(\frac{\lambda(S')}{T(S') \cdot C(S')}\right) \div \left(\frac{\lambda(S')}{T(S') \cdot C(S')}\right)$$

where $C(S')$ and $C(S)$ is the cost of achieving the states S and S', respectively. $(\lambda(S')) \div (T(S'))$ is the power associated with state S'. A system is said to be scalable from configuration S to S' if $\Psi(S, S') > 1$.

In a later paper [9], it is shown that this metric is not useful if response times approach zero, because it rewards very short response times even if they are smaller than the one required by the user. Therefore, a metric based on the cost-effectiveness or productivity is proposed, where the productivity is a function of the system’s throughput, its quality of service, and the running cost per second. This value function may be a function of any appropriate system measure, including delay (mean, variance, jitter, ...), availability, or the probability of data loss or time-outs.

2.3 Capacity Planning

2.3.1 Methodology

Fulfilling given system requirements is the objective of capacity planning. Capacity planing is based on models consisting of three parts [1]:

- The workload model captures the resource demands and workload intensity characteristics of the load brought to the system by the different types of transactions and requests.
- The performance model is used to predict response times, utilization, and throughput, as a function set of the system description and workload parameters.
- The cost model accounts for software, hardware, telecommunications, third-party services, and personal expenditures.

Models may be further sub-divided into natural models (real workload, traces) and artificial models, which are constructed from descriptive parameters. In the latter case, workload may either be characterized by graph, or it may be partitioned into several classes. The typical methodology of capacity planning is further detailed in Figure 2.3.

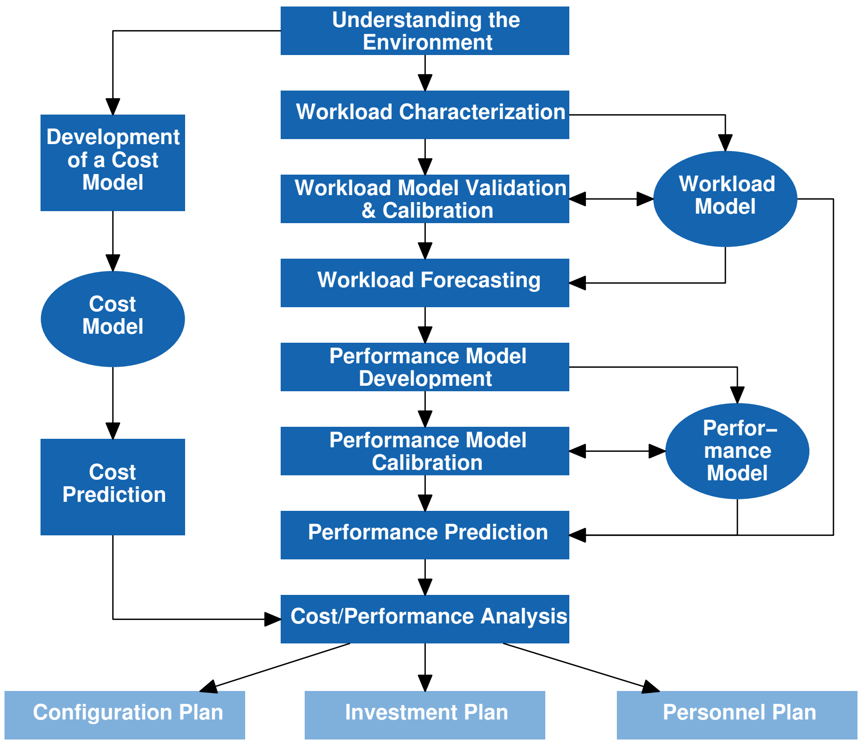

Understanding the Environment
Workload Characterization
Development of a Cost Model
Workload Model Validation & Calibration
Workload Model
Workload Forecasting
Cost Model
Performance Model Development
Performance Model Calibration
Perfor-mance Model
Cost Prediction
Performance Prediction
Cost/Performance Analysis
Configuration Plan
Investment Plan
Personnel Plan

Figure 2.3: Capacity planning methodology (from [1])

2.3.2 Workload Modeling

Workload modeling requires system models at different levels. There are different proposals for suitable levels. [1] suggest:

- User level (session layer)
- Application level (functional layer)
- Protocol level (HTTP requests)
- Network level

A slightly different model is proposed in [12], which illustrates that even more levels might be useful (see Figure 2.4).

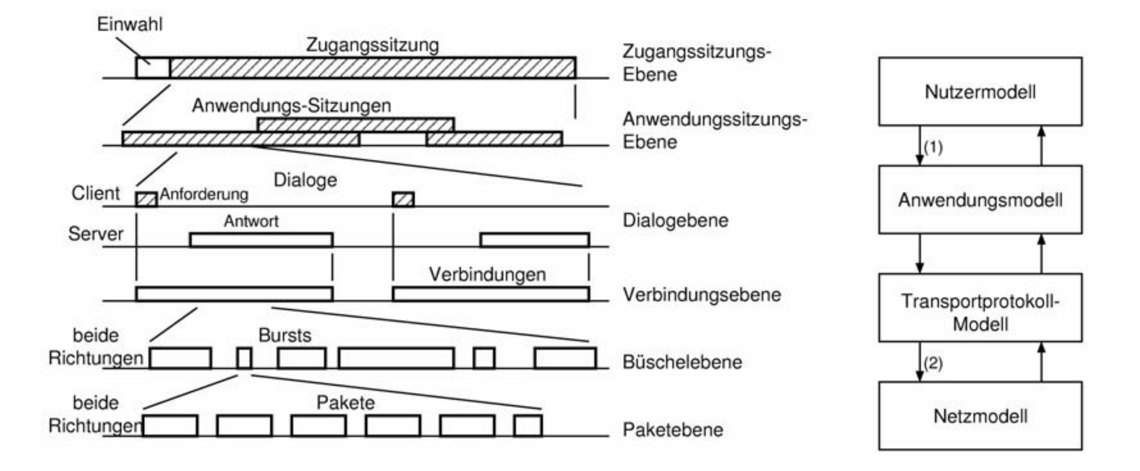


Figure 2.4: Workload modelling with different activity levels (taken from [12])

2.3.3 Mobility Modeling

User mobility has a very important impact on the performance of the Nexus system, in particular on mobility management components. Therefore, mobility has to be taken into account by capacity planning. A detailed discussion of different mobility modeling approaches in Nexus can be found in [13]. Therefore, this section only includes a brief overview.

Mobility models on a microscopic level assume the onset of vehicular or pedestrian traffic as a chain of cause and effects [14]. The fundamental assumption is that every traffic demand is motivated by basic human necessities like habitation, working, shopping, spare time etc. These necessities lead to activities that induce a need to change the location, i. e., the so-called individual (physical) traffic demand in the form of an activity sequence. Based on the activity sequence the place and the time of the activities and the mode of transportation are chosen. For the activities in defined area routes have to be selected (the criteria of the best route can be manifold). Finally, the (physical) traffic flow in form of the interaction between traffic participants takes places on the selected routes.

As a consequence, the chain of cause and effects can be roughly divided in the three parts:

- (physical) traffic demand
- (physical) traffic distribution
- (physical) traffic flow

While most small scale performance analysis work with the assumption of simple mobility models that focus on the (physical) traffic flow, in large scale scenarios the chain of mobility modeling should be considered. As described in [13], mobility models mimic the behavior in the real world. Such models can both provide *macroscopic* and *microscopic mobility metrics*. Important macroscopic metrics for vehicular traffic are the traffic volume, the mean car velocity, the car density, and sojourn densities. Microscopic metrics describe the mobility of single entities, such as the current speed, the travel time, the distance between different vehicles, etc.

The traffic demand - as the basis for performed mobility - is the most difficult part to capture. It does not have a physical representation, because not every demand can be fulfilled. Therefore, the typical method to capture traffic demand is the oral or written census of households that is obviously not objective. In Germany exists several censuses on different levels of detail, e. g., Kraftfahrzeugverkehr in Deutschland (KiD) [15], Mobilität in Deutschland (MiD) [16], Dateline [17], German Mobility Panel (MOP) [18], or the Mobility Panel for long-distance traffic (InVerMo) [19]. Basically one can distinguish between longitudinal or panel data (observation of a single person or a vehicle over the timeline) and cross-sectional data (observation of different persons with regard to a certain value at a point of time). While cross-sectional studies take a snapshot, the panel studies focus on the observation of processes (longitudinal observations).

Parameters of traffic demand (exemplified for the mobility panel) are for instance yearly collected data:

- Data relating to households (e. g., location, situation, moved household, number of persons, financial situation, number of vehicles, second home)
- Data relating to individual persons (e. g., ID, sex, age, graduation, profession, change of workplace, distance to workplace, anomalies, illness)
- Data relating to the trips (e. g., household, ID, day, weather conditions, intention, means of transportation)
- Data relating to the particular day (e. g., household, ID, day, illness, holidays, traveled distance, duration of travel, distance to different activities, mode)

These sources provide quite detailed information about vehicular mobility in Germany:

Table 2.2: Profile of German vehicular system

Metric	**Value**
Rural roads	230,848 km
among this: autobahn	11,786 km
Urban streets	413,000 km
Availability of passenger cars	69.3%
Participation in transport	92.1 % per day
Total amount of transport	718.6 billion km/a

There are some further characteristic metrics that describe a transport system, such as the share of persons participating in transport (in %), the number of kilometers per person and day (in km), the number of trips per person and day, the mean trip length per person and day (in km), or the travel time budget (in h).

Unfortunately, for pedestrian mobility only few information is available in the public. In order to obtain reasonable results, within the Nexus project a new pedestrian mobility census has been conducted at the campus of the Universität Stuttgart.

3 Nexus System

3.1 Location-based Services and Context-Aware Systems

Context-aware computing focuses on enriching applications with contextual information, like position, user activity, nearby people and devices, time of day, or weather conditions. There are different definitions for the terms *context* and *context-aware*. [20] provides the following definition:

> “Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.”
>
> “A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task.”

In general, there are quite a number of different context types, in particular location, identity, activity, and time. A very important class of context-aware systems are location-based systems providing location-based services (LBS). An LBS is a service for mobile users (terminals) where the awareness of the terminal of the current, past or future location forms an integral part of the service.

3.2 Overview over the Nexus System Architecture

The Nexus project research is motivated by the vision that future location-based services will be based on large-scale models of the real world, which are provided by many different providers and thus have to be federated by a web service middleware [21]. The system architecture of such a service platform is shown in Figure 3.1: A service requester, e. g., a mobile terminal, can perform requests to a middleware. This so-called *federation component* provides uniform access to information offered by different service providers. To announce their services, the providers publish descriptions to a service registry (*area service register*, ASR).

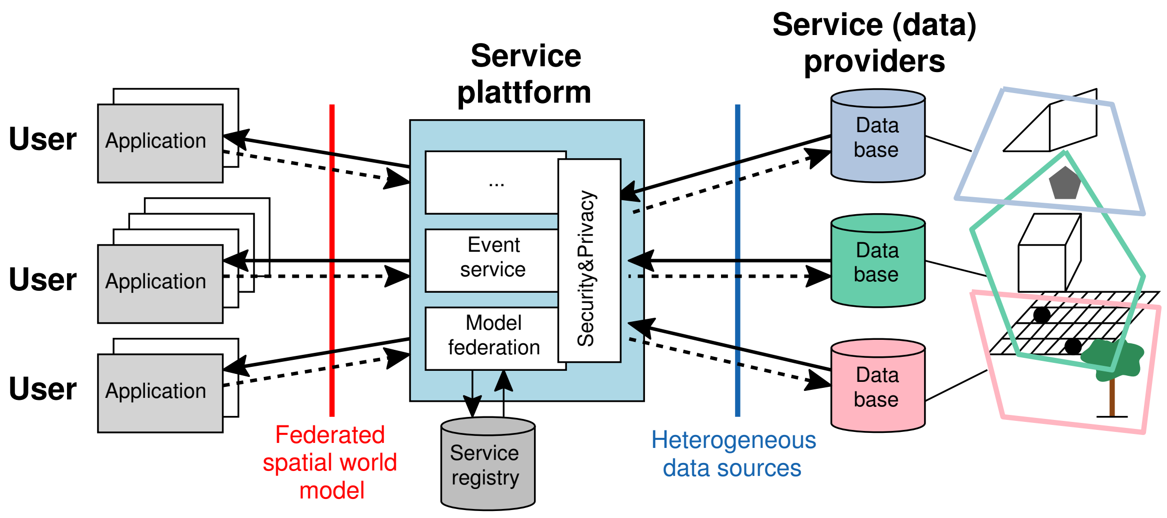


Figure 3.1: Nexus service platform architecture

As discussed in [21], such systems are unlikely to be realized by a single database if third-party content providers are involved. In the example of location-based services, this could mean that there are various providers offering information, e. g., about points of interest. The data is stored in databases and can be accessed by XML-based protocols (AWML, AWQL). When a request arrives, the federation component determines with help of the service registry which servers store corresponding data. Then, the request is decomposed into sub-queries to each server (fork). Thereafter, the partial results of the sub-queries are collected and integrated into a single consistent result (join), which is sent back to the client.

Unlike many other research projects, Nexus envisions that these service delivery platforms are open. This property can be defined as follows:

> “The defining characteristics of an open distributed systems are distribution and openness. Distribution entails that the system’s components reside at different physical locations. Openness means that components may enter or leave the system. Typically, openness is achieved by components having well-defined and published interfaces that support interactions through well-defined and published protocols.” [23]

The fact there are different databases operated by different service providers is one of the main differences of Nexus compared to other approaches, such as “Google Earth” [22], which uses a very centralized architecture. Such a centralized approach is not realistic for very detailed context models because of the following reasons:

- The architecture has to reflect organizational structures and administrative domains. Content providers are interested in having control over their data and are thus interested in operating their own servers.
- The locality of requests can be handled more efficiently if data is partitioned according to geographic locations. Furthermore, servers should be “close” to the requestors (in terms of network topology).

Further details on the technical realization of the Nexus platform are described in [24].

3.3 Potential Performance Bottlenecks in the Nexus System

As shown in Figure 3.2, the Nexus system consists of a (potentially high) number of nodes, which are interconnected by communication networks. Mobile Nexus terminals have access to the Nexus federation via mobile networks (UMTS, WLAN, etc.). The Nexus infrastructure itself typically communicates over wireline communication links.

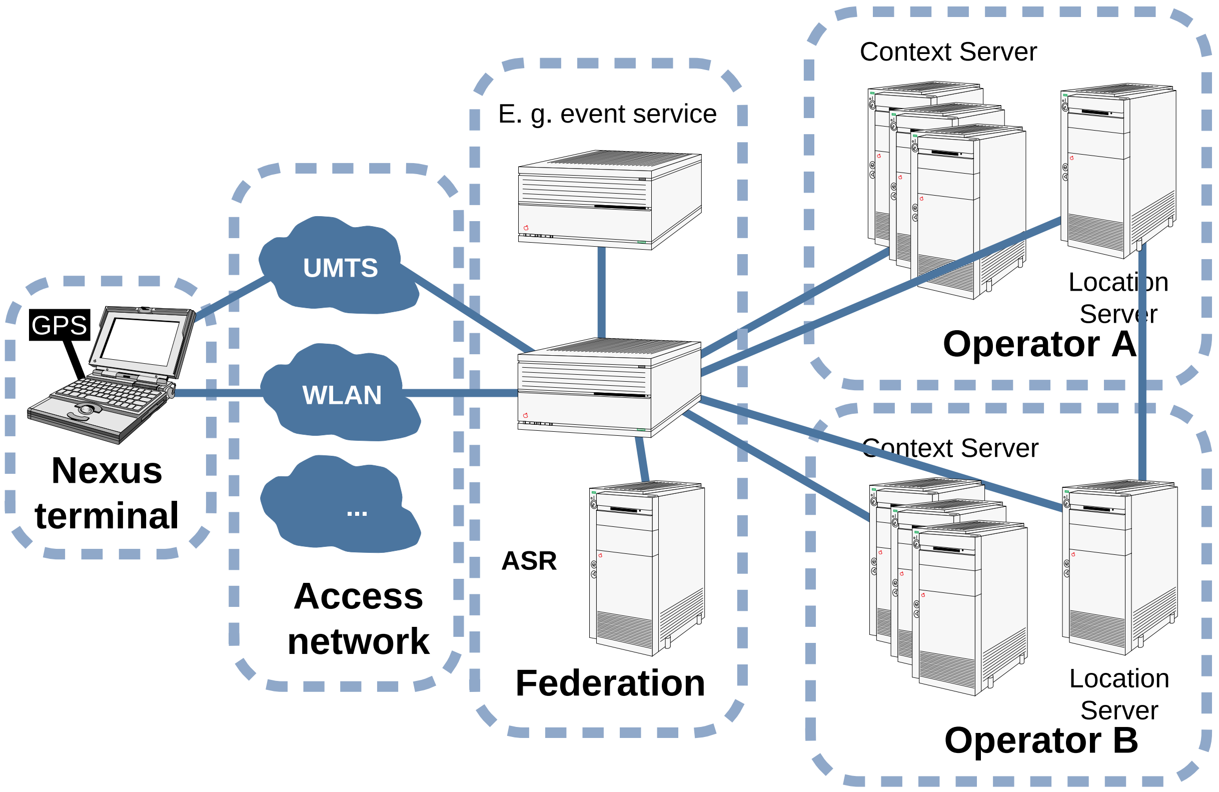


Figure 3.2: Typical scenario of the Nexus system

The total system performance of a complex system like Nexus depends on many factors, since the performance of each individual component makes a contribution. Particularly critical is the case that there are bottlenecks, i. e., certain components are overloaded. In database-like systems such as Nexus, overload typically results in an increase of the response time. In case of severe overload, situations may occur where request cannot be served at all.

Bottlenecks can occur at two different points in the Nexus system:

1. Nodes
   Nexus components such as the involved databases, federation components, etc. may not be able to process the number of requests. Potential reasons are:

   - Too many requests or responses,
   - Too complex processing tasks (e. g., federation of large-scale documents), or
   - Limited resources (such as energy).

   This is why central components have to be dimensioned so that they can handle a very large amount of operations.

2. Communication network
   All communication network links have a limited bandwidth. Both for local area networks (LANs) and wide area networks (WANs) technologies for high data rates exist (Gigabit/s and more). However, access networks, in particular wireless access networks, lack orders of magnitude behind these data rates. Network links can become a bottleneck due to:

- Very high amounts of data to be transmitted (both user data and signaling),
- Limited bandwidth at peering points of different network operators, or
- Usage of low-bandwidth communication technology, e. g., in sensor nodes.

The Nexus vision implies the usage of heterogeneous communication networks. The most important wireless access network technologies in Nexus are GPRS, UMTS, Wireless LAN, Bluetooth (e. g., in personal area networks), and, in future, potentially also WiMax. These networks have quite different characteristics, which can be summarized as follows:

Table 3.1: Characteristics of different access network technologies

	GPRS	**UMTS**	**IEEE 802.11 (WLAN)**	**Bluetooth**	**IEEE 802.16 (WiMax)**
Coverage	ubiquitous	cities	hot spots	person	cities
Cell size	several km	few km	100 m	1-10 m	several km
Data rate	30 - 50 kbit/s	384 kbit/s 1.8 Mbit/s with HSDPA	5 - 7 Mbit/s, or more	up to 1 Mbit/s per piconet	up to 70 Mbit/s
RTT	500 ms - 3 s	200 - 300 ms	5 - 40 ms	ca. 50 ms	small
Mobility	link layer	link layer	IP layer or above	ad hoc	IP layer or above
Deployment cost	high	high	low	low	medium
Cost per bit	high	medium	gratis ... high	-	unknown

4 Building Scalable Systems

Much work dealing with the scalability of distributed system has been done in the context of the World Wide Web (WWW). Considering the fact that the Area Service Register (ASR) in Nexus has an equivalent function like the domain name system (DNS), and that context servers are somewhat similar to Web servers, many strategies used to provide services in the WWW in a scalable way can be used for the Nexus system, too.

4.1 Scalability Strategies in the WWW

Web servers contribute for about 40% of the delay in a Web transaction (cf. [8]). This highlights the importance to implement the server in an effective and efficient way. The ability to scale up a system may depend on the types of data structures and algorithms used to implement it or the mechanisms its components use to communicate with one another [11]. In particular, the important factors are the CPU, the disk, the memory, and the network [8].

[7] identifies two different strategies to improve the scalability of a web site:

- *Scale-up* strategy: The system is expanded incrementally by adding more devices to the existing node(s). In practice, this means that servers are replaces by better (faster) servers, but their number remains constant. In the simplest case, only one server is deployed (single node).
- *Scale-out* strategy: The system is expanded adding more nodes, i. e., extra servers. As a result, the system consists of multiple nodes.

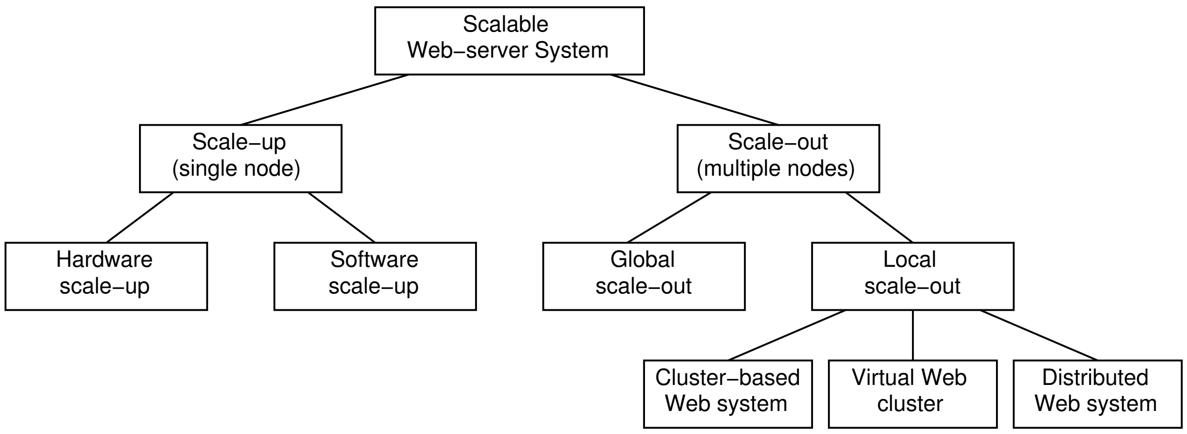


Figure 4.1: Architecture solutions for scalable Web-server systems (from [8])

A detailed overview over different approaches to build scalable web servers can be found in [8]. The main design alternatives are illustrated in Figure 4.1. With respect to the scale-up approach, one can distinguish between *hardware scale-up* and *software scale up*. In the former case, a system is expanded by adding hardware, e. g., a faster processor, whereas in the latter case the performance is improved at the software level, e. g., by optimizing the operating system. *Scale-out* approaches are used in cluster-based systems, in particular in Web server farms. Possible architectures for such server farms include cluster-based Web systems, virtual Web clusters, and distributed web systems. Basically, they differ in the usage and visibility of IP addresses. The main components of a cluster are

- a request routing mechanism to direct a client request to a target server node (content-blind routing vs. content-aware routing),
- a dispatching algorithm to select the node that is considered best suited to respond (load balancing vs. load sharing, centralized vs. distributed, static vs. dynamic), and
- an executor to carry out the dispatching algorithm.

An overview of different data management techniques in a server farm is given in [7]. Basically, there are two design choices:

- *Cloning* or *replication* of services: In a so-called Reliable Array of Cloned Services (RACS), all replica clones have the same software and data. Arriving requests are processed by one of the clones so that a load balancing is achieved. Cloning has several advantages: The scalability and availability, i. e., the fraction of the presented requests that a system services within the requested response time, can be improved. Clone failures can be completely masked if node and application failure detection is integrated with the load balancing system. [7] further distinguishes between *shared-disk clones* (or clusters), i. e., stateless servers accessing a common back-end storage server, and *share-nothing clones*. In the latter case, each clone has all data stored, which is simpler to implement and scales better for read-only applications. For large or update-intensive databases shared-disk clones are more economical.
- *Partitioning*: In a Reliable Array of Partitioned Services (RAPS), the data is divided among the nodes. Unlike mirroring, this partioning is transparent to applications. Each request is directed to the appropriate partition and processed by this partition. In this case, the load balancing is more complex, and the availability is not improved if the data is stored in only one place. Therefore, partitions are usually implemented as a *pack* of two or more nodes that provide access to the storage. RAPS may either consists of *shared nothing packs* or *shared-disk packs*.[1]

4.2 Caching and Content Delivery Networks

The most important mechanism to enhance performance is *caching*. Caching improves network and system performance by saving network bandwidth, reducing delays to end clients, and alleviating server load. Caches can (and should) be deployed at different scales. One must differentiate between internal caches inside the Web server system and external caches, often realized as proxies. A study on the question where to place transparent en-route caches is presented for instance in [25].

Important considerations for caching are:

- What, when, and where to cache,
- the granularity of caching (web pages, fragments of pages, query results),
- the location of cache (client, proxy, edge, Internet service provider, application server, data base management system, ...), and
- the caching and invalidation policies.

1. The shared-disk pack is virtually identical to a shared-disk clone, except that the pack is serving just one part of the total database.

There are different types of caches: Fragment caches, command caches, query results caches, database caches, etc. The effectiveness of caching can by measured in terms of the hit ratio, the byte hit ratio, and the data transferred.

Caching is the underlying mechanism of so-called content delivery networks (CDNs), which are used to provide Web sites at a global scale. Commercially CDNs rely on proprietary mechanisms that are hardly documented. [8] provides a comprehensive survey over architectures consisting of multiple server nodes distributed on a local area, with one or more mechanisms to spread client requests among the nodes. It focuses on architectures, internal routing mechanisms, and dispatching request algorithms for designing and implementing scalable Web-server systems under the control of one content provider.

It is important to note that caching is only effective for rather static data. For dynamically generated data, caching does not improve performance, or may cause problems due to inconsistencies.

4.3 Scalability of Existing Web Portals

Due to the tremendous success story of the WWW, the access rates on certain services are very high. For instance, [26] gives some information about the order of magnitude of traffic for some important service providers:

Table 4.1: Known workloads in the WWW (according to [26])

Site	**Requests/d**	**Req./s (mean)**
AOL Web cache	10 billion	120,000
Inktomi search engine	80 mio	930
Geocities	25 mio	290
Web-based email	1 billion	12,000

The data of this reference is of 2001. Since the number of online users increases, today’s figures can be assumed to be even higher. Furthermore, the popularity of e-commerce web sites such as “ebay” or search engines such as “Google” has very much increased in the last couple of years.

In general, there is not much information available about the workload of today’s WWW infrastructure. This is also noticed in [27]: “There are very few published studies of e-business workloads because of the difficulty in obtaining actual logs from electronic companies. Most companies consider Web logs to be very sensitive data.”

It is well-known that the workload on a Web site is subject to large fluctuations during one day, and also during the week. This can be seen for instance from some figures published in [27]:

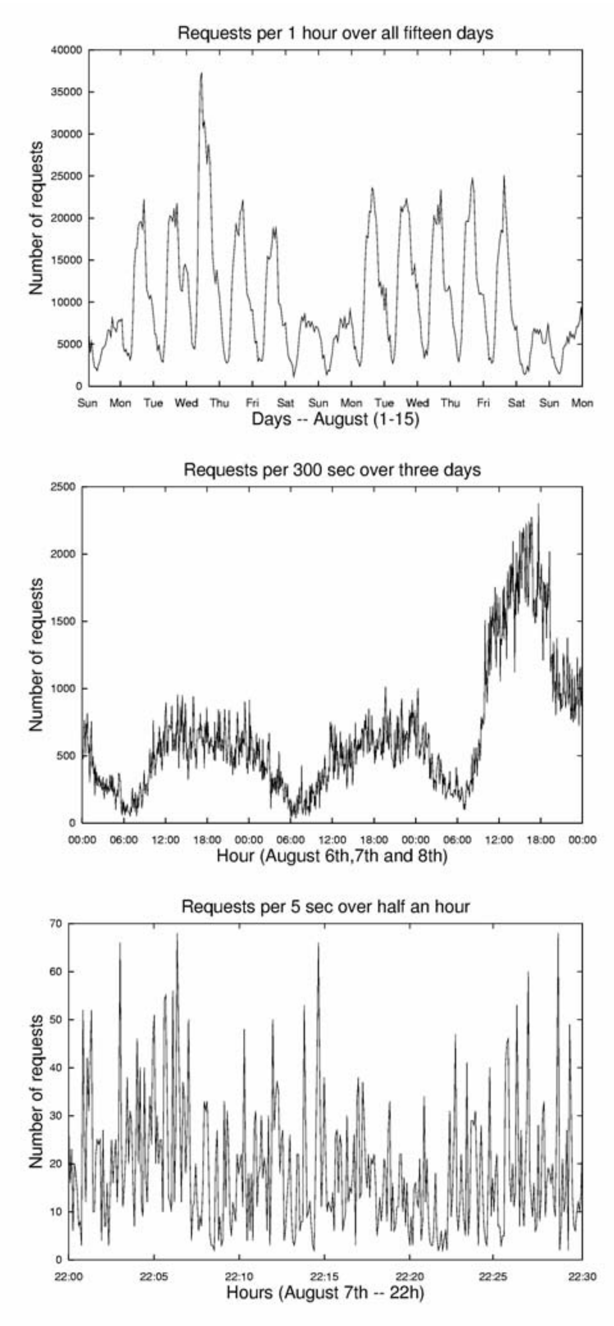


Figure 4.2: Workload of an online book shop / e-commerce platform (from [27])

4.4 Scalability of Existing Location-based Service Platforms

Many cellular network providers offer location based services. The underlying location data infrastructure is illustrated in Figure 4.3: It consists of a location measurement unit (LMU) and the serving mobile location center (SMLC), which processes measurement data, calculates positions, checks access rights and capabilities, and which is also involved in accounting and billing. Further important components are the home location register (HLR) storing user data and the current location (in rough granularity), and the gateway mobile location center (GMLC), which is the interface to entities outside of the mobile network.

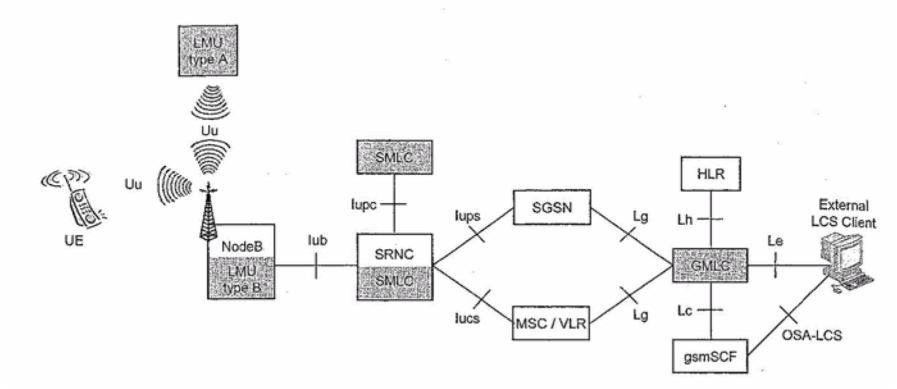


Figure 4.3: Location-based service architecture in GSM (source: [3])

The HLR is the central database in this architecture. Today's HLR databases are based on modern IT technology. Some data on their performance is given in [28]:

- Authentication for > 100 million devices
- Mean transaction rate of 100,000 tps, corresponding to 4 Gbits/s
- Short response times of < 100 ms (This is a requirement to keep call setup delays small.)
- High availability

The increasing popularity of location based services leads to an increase of the signaling traffic in mobile network. This issue has been investigated within project A1 [29]. One of the results is that most existing signaling links can transport this additional traffic, while the additional load on HLRs may become critical when the number of HLR queries per user increase very much. This is also illustrated for an example result in Figure 4.4.

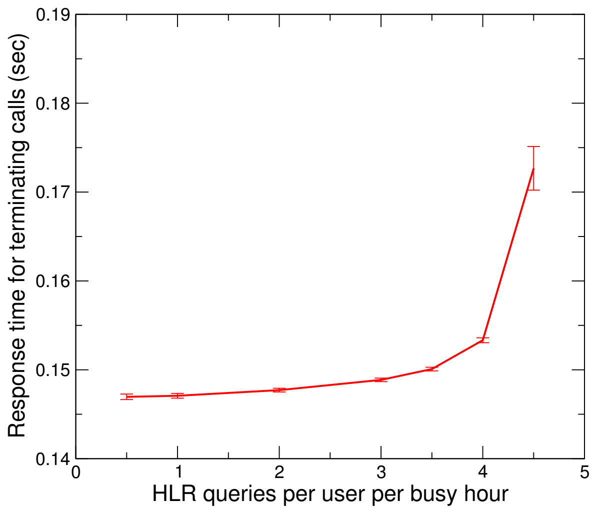


Figure 4.4: Example result of HLR performance (taken from [29])

The location-based services themselves are usually offered by third-party providers. Unfortunately, these companies hardly publish data that provides information about the scalability of there platforms. Some exceptions are:

- Microsoft’s Map-Point Service [30]
  According to the web site, this service manages 200,000 points of interest and 15 Mio adverts in 2005. Guaranteed service level agreements (SLAs) are 1 second response time (in the data center) and 99,9% availability.

- IntelliWhere Location Server [31]
  According to the web site, this infrastructure can deliver at least 24 Mio maps per day, i. e., 28 per second.

4.5 Applicability to the Nexus System

The Nexus System has some similarities to the WWW, in particular with respect to the openness of the architecture. Therefore, one can expect that Nexus services may have similar characteristics as many Web-based services have today, such as “google”, “ebay”, etc.

However, it should be noted that there are three major differences to most existing systems:

1. Federation
   The concept of federating many different and heterogeneous data sources is not used by existing online service platforms, which are usually a single domain solution. There is ongoing research in several related research projects, but there are still many problems to be solved.

2. Large amounts of data
   Unlike Web sites, world model information can be very voluminous. This can be seen from two example scenarios: The NanoCamp data, which mainly contains two-dimensional representations of buildings and streets in the city center of Stuttgart, required about 7 MB for 26,000 objects, and a detailed model of the computer science building on the Vaihingen campus has a size of about 12-15 Mbyte (if modelled in GML).

3. Dynamic information
   The data management of world models is particularly challenging since world model information can be very dynamic. Existing content delivery networks show that it is possible to handle very large amounts of data, as long as they are rather static and thus can be cached. In Nexus, however, both the location of objects can be variable (i. e., mobile objects), as well as attributes (e. g., sensor information). Partial solutions to this problem have been developed in Nexus: For example, it is possible to build a scalable location service [32] if there is no federation of different providers, i. e., the service areas of different servers do not overlap. For the federated case, different solution options exist [33].

5 Nexus Performance Evaluations

This section gives a brief overview of studies on the performance of the Nexus system, which have been performed in different sub-projects of the center of excellence.

5.1 Evaluation of the Federation Concept

The Nexus world model is based on information stored in distributed databases that are offered by different service providers. The federation component provides uniform access to these multiple data sources. In order to handle a request, this component must query different service providers in parallel and integrate their responses into a single result. In the context of web services, this mechanism is an example of service *composition* [34]. The same principle is known as data *mediation* in distributed databases. Service composition is based on the fork-join computing paradigm [35]: A request, or generally speaking a job, is distributed over several service units and can only be finished when all of them have completed processing. Thus, the overall response time includes a synchronization delay that is determined by the slowest service unit.

In [6], project A1 has studied the response time of a such a federation component. The work is similar to the paper [34] that analyzes the effects of exponential response times based on an earlier work in [36]. Unlike these studies, [6] explicitly considers the effort of joining results. Detailed queuing models for distributed databases have also been studied extensively, e. g., in [37]. However, most existing work only considers constant or exponential service times, while measurements in the WWW and in e-commerce systems have observed heavy-tailed server response time distributions [38] [39]. From this follows that this effect is likely to occur in the Nexus system platform, too.

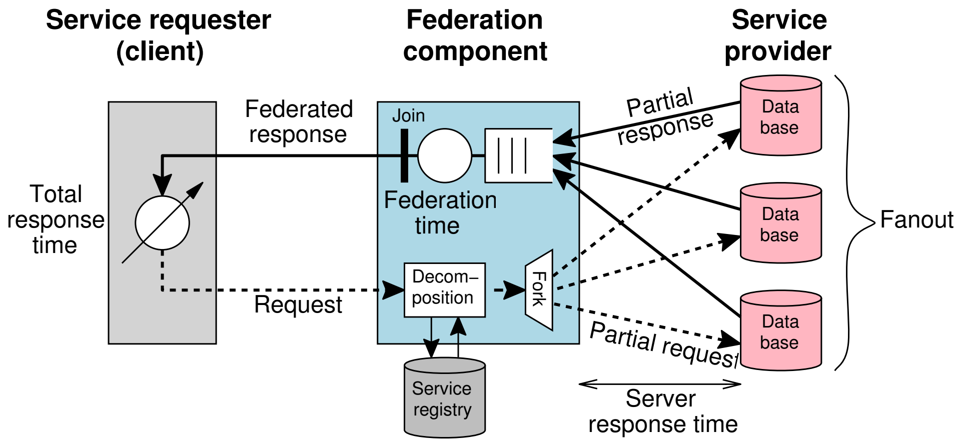
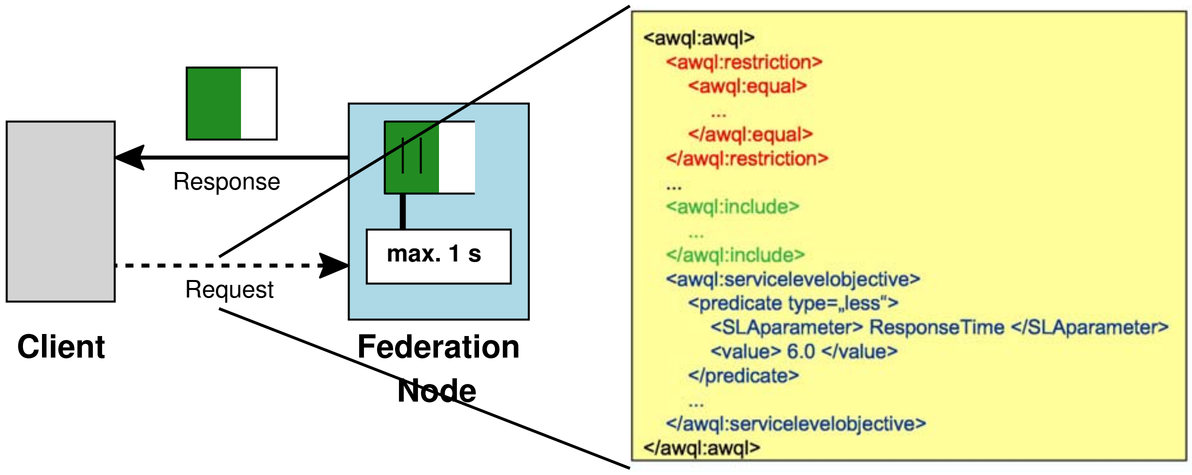


Figure 5.1: Queueing model used in [6]

This raises the question how scalable the Nexus federation concept is, in particular, if the number of content providers increases and thus many servers have to be queried. The number of servers invoked in parallel is labeled as the “fanout factor” in [6]. The main performance metric of the Nexus system platform is the response time of the federation component that is composed of two main sources of delay: First, the response times of the databases, which also include processing and transmission delays. The second main source of delays is the federation component, where each incoming response has to be processed and merged to the overall response. For complex data structures this join operation can require significant processing. Thus, the Nexus federation can be modeled by the queueing model shown in Figure 5.1.

If server response times are heavy-tailed, i. e., if single values are likely to be orders of magnitudes higher than the mean, the federation response time can be very large. The reason are synchronization delays: The federation has to await all servers, even those that are very slow. The paper [6] quantifies this effect and shows that it is difficult to provide a federation service if server response times are heavy-tailed, as in the WWW.



```
<awql:awql>
  <awql:restriction>
    <awql:equal>
      ...
    </awql:equal>
  </awql:restriction>
  ...
  <awql:include>
    ...
  </awql:include>
  <awql:servicelevelobjective>
    <predicate type=„less">
      <SLAparameter> ResponseTime </SLAparameter>
      <value> 6.0 </value>
    </predicate>
    ...
  </awql:servicelevelobjective>
</awql:awql>
```

Figure 5.2: Illustration of first approach: Setting a timeout in request

The paper [6] proposes and analyzes two strategies to reduce the response time: In the first approach, the federation component does not wait for slow servers, i. e., it only considers responses of a part of the databases. For instance, it might be sufficient if Nexus provides information about *most* points of interest in a certain area. Thus, if some results can only be obtained by waiting for responses from very slow servers, it might be better to construct the federated response earlier. As shown in Figure 5.2, this could be realized by a timeout. When this timeout expires, the federation component does not further wait for servers. Alternatively, long response times can be addressed at the servers by a monitoring component that sends an error message if no result can be provided within a certain time. Both solutions imply a tradeoff between performance and response completeness. Analytical and simulation (such as the ones depicted in Figure 5.3) show that omitting a few slow servers (such as 5%) or setting a timeout (e. g. of 10 s) can significantly improve the mean response time of the Nexus system platform.

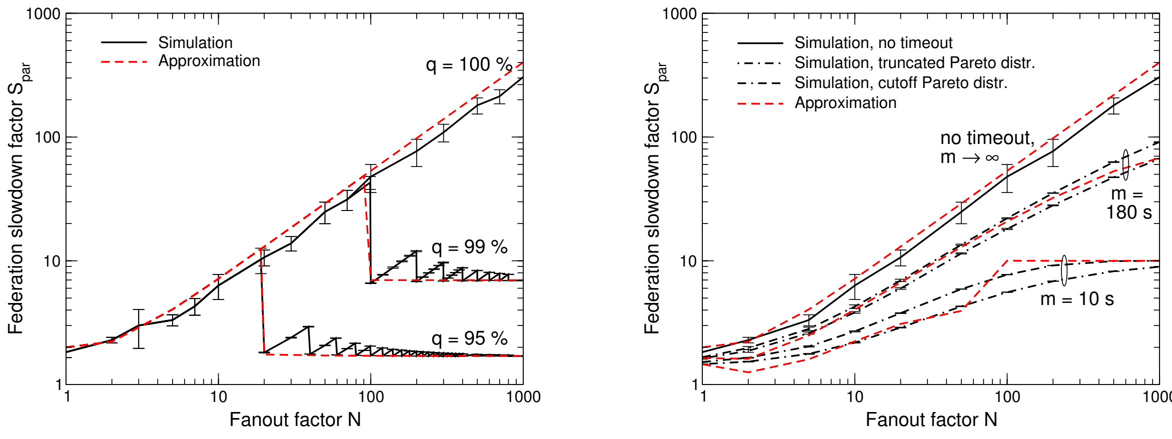


Figure 5.3: Example analytical and simulation results for restriction to fast servers (left) and server-based timeouts (right)

5.2 Performance of Blind People Navigation Support System

The Taskforce Performance has analyzed the question whether the Nexus application example of blind people navigation support can be realized using cellular access networks. For this analysis, some assumptions have to be made:

- According to [40], there are about 155,000 blind people in Germany. From this follows a number of about 1000 persons for the city of Stuttgart.
- The Nexus system offering online information could be realized as follows: A caching component in the end device stores all model information within a range of (at least) 100 m. If a blind pedestrian walks with a velocity of 0.5 m/s (normal: 1.4 m/s) in a straight line, approximately every 200 s a new data set has to be downloaded.
- The Nexus world model has to offer static data such as obstacles, curbs, undercrossings, traffic lights, bus stops, shop entry doors, etc. A typical object can have a size of about 100 bytes. Assuming one object per square meter, 10,000 objects or 1 Mbyte has to be downloaded every 200 s. From this follows a mean data rate of 40 kbit/s. Of course, this data rate can be drastically reduced if static data is stored offline, e. g., on a DVD.
- Further dynamic data such as information concerning navigation, time tables, opening hours, state of traffic lights, events, etc. can cause further traffic. Assuming again 100 byte/object, and a mean number of one object to be transmitted per second, the mean data rate is 0.8 kbit/s.

The resulting data rate of about 41 kbit/s can still be realized by GPRS and can easily be provided by UMTS or HSDPA, provided that not too much other users are communicating in parallel in the same cell. From this follows that this application scenario is realistic and can be realized with state-of-the-art network technologies.

5.3 Federated Nearest Neighbor Queries in Nexus

In [41], a family of algorithms for processing nearest neighbor (NN) queries is proposed for the Nexus platform. Previous approaches for parallel and distributed NN queries considered all data sources as relevant, or determined the relevant ones in a single step by exploiting additional knowledge on object counts per data source. The approach presented in [41] that does not require such detailed statistics about the distribution of the data. It iteratively enlarges and shrinks the set of relevant data sources. Experiments show that this yields considerable performance benefits with regard to both response time and effort. Additionally, it is proposed to use only moderate parallelism instead of querying all relevant data sources at the same time. This allows to trade a slightly increased response time for a lot less effort, hence maximizing the cost profit ratio.

6 Scenarios

6.1 Motivation

The following subchapters include a list of scenarios that can be used to evaluate the scalability of the Nexus system platform. It has to be emphasized that these scenarios are focusing on large-scale application scenarios that require an efficient design of the system platform. A common characteristic of such scenarios is that they have requirements that are partially contradicting: As illustrated in Figure 6.1, it is difficult to build a generic and flexible system that handles dynamic information in a scalable way. However, any combination of two of three requirements can be realized with rather simple means.

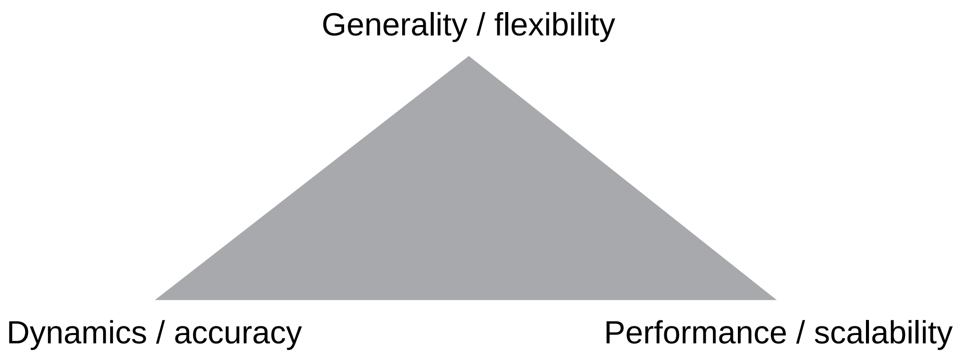


Figure 6.1: Three requirements on systems that can hardly be fulfilled in parallel

It must be mentioned that the scenarios in this document are not identical to the application examples used for demonstration. Small scenarios of limited complexity, e. g., indoor-only scenarios such as the SmartRoom, are not included in this list since they have only moderate requirements concerning performance.

Of particular interest is the structure of the data that has to be stored in the context models. Different aspects have to be considered:

- Number of objects: few objects or many objects
- Origin of information: single source or federation of various sources
- Spatial extension: country-wide, state, city, single object
- Type of information: simple objects, complex objects

As will be shown in the following sections, these characteristics are specific to the application scenario.

6.2 Classification

In the following, different Nexus application scenarios are listed and classified according to four categories:

1. Mobile devices and access networks
   - Devices: Mobile phone, smartphone, PDA, laptop, desktop PC, ...
   - Access networks: GSM, UMTS, WLAN, Bluetooth, ...

  - Location method: GPS/Galileo, WLAN-based, GSM-based, in-door positioning, RFID, ...

2. Users and applications
   - Content: Information the system deals with
   - Number of users
   - Type of users / mobility: Pedestrian, vehicular, ...
   - User behavior
3. Quality of Context requirements
   - Precision
   - Correctness
   - Trust-worthiness
   - Resolution
   - Up-to-dateness
   - QoS
4. Nexus Services
   - Number of operators: one, several, or many
   - Nexus infrastructure: Amount of servers and distribution of content, dynamics, complexity
   - Nexus components: What parts of the Nexus system architecture are required in the scenario.

Finally, performance issues are discussed for each scenario.

6.3 List of Scenarios

6.3.1 Scenario 1: Tourism/Campus Information System

Description

A campus information system offers location-based information on a University campus. A tourism information system in a city providing local information has very similar characteristics.

- Different institutes operate Context Servers and offer location-based information. There could be multiple representations.
- Example use cases:
  - Searching the next copying machine
  - Searching a certain lecture hall
  - Searching a free meeting room
  - Navigation
  - Location-based events

Such a system could be particularly beneficial when the semester starts and many new students arrive at the Campus.

Characteristics

Mobile devices and access networks	**Users and applications**	**Quality of context requirements**	**Nexus services**
Device: PDA or similar Access network: WLAN Location method: WLAN, GPS, RFID	Content: Rather static WWW-like information, one for each site of interest (max. 1000). Each page consists of text (1 kbyte-10 kbyte) and pictures (100 kbyte-1 Mbyte) Number of users: 100-1000 Type of users/mobility: pedestrians in city center Usage behavior: walking around and using services from time to time (e. g. every 5 min)	Precision: low (10-100m) Correctness: medium Trust-worthiness: low/medium Resolution: high Up-to-dateness: low (10min-1h) QoS: Response time up to few seconds	Number of operators: one (e. g., city, university), closed system Nexus infrastructure: centralized, one server Nexus components: Context Server, Location Service, Event Service, Infostations, context-based information display on terminal

Performance Analysis

System load:

- Total number of requests: 1000 User*1 Request/User/5 min = 3.33 Request/s
- Total data rate: 3.33 Request/s * 100 kbyte/Request = 333 kbyte/s = 2.67 Mbit/s

This load can be handled with a single context server. The wireless network access would only be critical if all users used a single or very few UMTS cells.

Conclusion: Probably no severe performance problems, existing prototypes, almost state-of-the-art.

6.3.2 Scenario 2: Friendfinder-/Dating-Service

Description

Find persons (e. g., your friends), get notifications if someone is close to your current position. Alternatively, service for parents to track children.

Characteristics

Mobile devices and access networks	**Users and applications**	**Quality of context requirements**	**Nexus services**
Device: mobile phone or smartphone, or PDA Network access: GSM, UMTS Location method: GSM, UMTS, maybe GPS	Content: very dynamic location information of persons Number of users: In principle, all GSM-subscribers (1-2 billion users). In Europe, the scope of a system will probably be national, with a typical user age of 14 to 25. In Germany, this would result in at most 10 Mio potential users. More realistically, one can assume 2-3 Mio users, i. e., about 1 Mio users for one large service provider Type of users/mobility: from stationary to very mobile Usage behavior: Users query the position of their friends, e. g. in the evenings. Peak load: 1 req/user/hour. A small number of users also registers location-based events ("if my friend arrives...")	Precision: low (10m-100m) Correctness: high Trust-worthiness: high Resolution: medium Up-to-dateness: medium (1min-10min) QoS: Response time up to a few seconds	Number of operators: Heterogeneous: Several (competing) operators, such as GMX and web.de, or mobile network operators (T-Mobile, Vodafone, ...). Maybe also smaller providers, non-commercial providers or even private persons. Nexus infrastructure: Distributed between the different operators. The internal structure within the domain of one operator, e. g. the number of servers, is chosen to meet the performance requirements. Nexus components: Location Service, Federation, Event Service

Performance Analysis

- There are three different traffic types: Location updates (for every user every 1-10min, can be optimized e. g. by dead-reckoning methods), queries, and event signaling.
- Operators typically offer the service at national range, i. e., the service areas overlap completely. As a consequence, range queries might always have to be forwarded to all operators. Therefore, the gateways between service providers might easily become a bottleneck.

- This service requires privacy protection such as queries to certificate servers and encryption and signatures both in servers and in terminals. This further increases the signaling load.
- System load:
  - Peak location update rate for one provider: 1,000,000 Users * 1 Update/10min = 1600 Updates/s (or higher, if more frequent updates)
  - Peak query rate for one provider: 1,000,000 Users * 1 Query/1h = 270 Queries/s
  - Event signaling: Depends on architecture

Conclusion: Signaling in the location service and in the federation is critical due to the potentially high number of users.

6.3.3 Scenario 3: Location-based Advertisement Service

Description

Advertisements in a shop a distributed by Geocast messages. For instance, messages can be sent to the electronics section of the shop. One message consists e. g. of 1000 byte text and a 10 kb image. Messages are sent every minute. Clients receive messages via their mobile phone or their PDA connected to the shop’s own WLAN hotspot.

Characteristics

Mobile devices and access networks	**Users and applications**	**Quality of context requirements**	**Nexus services**
Device: mobile phones, PDAs Network access: UMTS, WLAN Location method: IR, RFID	Content: positions of mobile users, coverage of access networks/cells Number of users: millions of potential users; only tens of "active" users receiving message Type of users/mobility: pedestrians Usage behavior: users are in general passive, walking around, receive advertisements; might request for further information	Precision: medium (~10 m) Correctness: low Trust-worthiness: low Resolution: low Up-to-dateness: (1min) QoS: Message delivery delay less than a second	Number of operators: senders: many (one for each shop); few carriers/operators for message distribution Nexus infrastructure: Distributed Geocast Nexus components: Distributed Geocast infrastructure, Context Servers (to define target area), possibly Location Service

Performance Analysis

If we assume a large number of shops per cell, the cell load can be high. Example: 50 shop per cell. Load: 50 x 10 kbyte / 60 s ~ 68 kbit/s if messages have to be sent only once within a cell (broadcast/mulicast); otherwise we have 68 kbit/s per receiver.

Conclusion: When the air interface offers a point-to-point message delivery only, offering this kind of service may be difficult.

6.3.4 Scenario 4: Disaster Warning System

Description

The Disaster Warning System distributes warning messages to all people in the endangered area.

Example: Fire in a chemical factory
About 10,000 people close to the fire are to be warned. A warning message consists of a short text message of about 1000 byte. This message is repeated every 5 min.

Characteristics

Mobile devices and access networks	**Users and applications**	**Quality of context requirements**	**Nexus services**
Device: mobile phones Network access: UMTS Location method: GPS, cell id, ...	Content: coverage of access network (cells) Number of users: millions of potential users; hopefully only at most 10,000 "active" users in the endangered area Type of users/mobility: pedestrians, cars, trains, busses, etc. Usage behavior: Users are passive, walking and driving around, receive message from system	Precision: low/ medium (~50m) Correctness: medium Trust-worthiness: high Resolution: low Up-to-dateness: medium/high (1min) QoS: Message delivery time of the order of seconds	Number of operators: senders: police, fire departments; ~5 carriers/operators for message distribution Nexus infrastructure: Distributed Geocast Nexus components: distributed Geocast infrastructure, Context Servers (to define target area; closed system sufficient), possibly Location Service

Performance Analysis

Conclusion: The very bursty traffic with large numbers of receivers/cells can lead to bottlenecks in the Geocast infrastructure.

6.3.5 Scenario 5: Blind People Navigation Support

Description

Blind people navigation support. This scenario can subdivided into two aspects:

- (Add-on) Information offers for blind people (such as opening hours, public transport system, etc.)
- Navigation for blind people

It must be mentioned that the latter scenario is much more challenging due to various reasons, e. g., concerning the required quality of context.

Characteristics

Mobile devices and access networks	**Users and applications**	**Quality of context requirements**	**Nexus services**
Device: special (pointing) device Network access: high availability required: GPRS/UMTS, maybe WLAN, too Location method: GPS, indoor positioning	Content: Static and dynamic information such as timetables, highly dynamic sensor information (e. g. traffic lights), many mobile objects (e. g. busses), doors (open/closed) Number of users: 10-100 in a city center Type of users/mobility: pedestrian Usage behavior: Walking around, going shopping, etc. In order to guide the blind person, the system automatically monitors the information about the surrounding contained in world models. Dangerous situations will trigger an event.	Precision: high (10cm-1m) Correctness: very high Trust-worthiness: very high Resolution: very high Up-to-dateness: very high (1s-10s) QoS: Response time of the order of several hundred milliseconds	Number of operators: large-scale system with many federated service providers in order to obtain all data needed Nexus infrastructure: Many very heterogeneous components Nexus components: Federation, Context Server, Location Service, Event Service

Performance Analysis

In principle, both data transmission in the radio access network as well as data processing and signaling in the platform could become a bottleneck. However, due to the low number of users and a limitation of the usage to city centers, where radio cells are densely deployed, the data transport should be feasible.

However, the scenario is very challenging because of other reasons: It is unclear whether the required amount of data can be gathered. For performance, signaling load in platform will be very high, and application has strict QoS and QoC-requirements. Static information such as map data is not critical since it can be stored offline (on a DVD). The main challenge is dynamic information and mobility. For instance, the system must determine every second whether something relevant has changes, by checking the state of all objects and sensors (including mobile objects) within a distance of 100 m.

Conclusion: Requires a very scalable system.

6.3.6 Scenario 6: Smart Factory

Description

Production resource management system, e. g., for management of mobile tools (milling, drilling, etc.). A more detailed description can be found in [42].

Characteristics

Mobile devices and access networks	**Users and applications**	**Quality of context requirements**	**Nexus services**
Device: mobile resources (RFID) Network access: WLAN, maybe Bluetooth Location method: RFID, indoor positioning system	Content: detailed static information about machines and tools, dynamic information such as abrasion, location, ... Number of devices: 400 machines, 200,000 mobile resource tools, ca. 92% (184,000) stored, 8% (16,000) used Type of users/mobility: movement between storage and machines Usage behavior: location updates for objects all 10 s to 30 s (could probably be reduced by optimization techniques)	Precision: high (15 cm) for tools that are currently in use, but probably only locally Correctness: high Trust-worthiness: - (single provider) Resolution: high Up-to-dateness: high (10-30 s) QoS: Application-dependent	Number of operators: one company, closed system Nexus infrastructure: Probably rather centralized system Nexus components: Location Service, Context Server, Event Service?

Performance Analysis

According to [42], the signaling load e. g. in the location service is quite high (> 1000 updates/s). But since most of the devices are not used, the traffic can be significantly reduced by more intelligent protocols. Being a closed in-house scenario, bandwidth is not as scarce as in public wireless networks.

Conclusion: Requires careful system design.

6.3.7 Scenario 7: Telematics

Description

Real-Time Traffic Context Information System (fine-granular traffic sensor network)

As an example of a fine-granular sensor network, one could assume a collection of all sensor information arising in the context of roads that have influence on the traffic flow. For example, information measured in a lot of modern vehicles like rain, temperature, but also emergency brakes or other sudden behavior of drivers. One could also assume more detailed sensor information like velocity and acceleration. Further information are collected via stationary sensors next to roads (fog, traffic volume, ...).

This precise and actual context information is offered to drivers, combined with information about points of interest (fueling stations, restaurants, etc.). Further services could include:

- Dynamic routing updates dependent on the traffic situation (event service)
- Location-restricted traffic jam advertisements (Geocast)
- Management of position of vehicles (fleet management)
- Passenger entertainment with world model information
- Vehicle-to-vehicle communication (ad-hoc), e. g., for information about road state
- Search for witnesses after accidents
- Radar speed check information service (legally critical)

Most of these services require the federation of information of different sources.

Characteristics

Mobile devices and access networks	**Users and applications**	**Quality of context requirements**	**Nexus services**
Device: the vehicles sending and receiving information Network access: wireless and wireline sensor network Location method: current toll-collect system (relative positioning via the distance to reference points), GPS	Content: highly up-to-date information Number of users: potentially all cars in a country, plus transit traffic, in Germany about 53 Mio; in reality all cars on the roads in a certain area Type of users/mobility: vehicular, at speeds from 30 km/h to 160 km/h Usage behavior: Drivers get important information of sudden events that were send by other drivers (ad hoc network) or a provider.	Precision: low (10m-100m) to medium (1-10m) Correctness: high Trust-worthiness: very high Resolution: low to high (depending on traffic density) Up-to-dateness: very high (1s-10s) QoS: Information that are send to drivers must be correct and in emergency case highly up-to-date.	Number of operators: one, or more Nexus infrastructure: Distributed, may include ad-hoc networks Nexus components: Sensor Context Server, Location Service, Event Service/ Geocast, Mobile Federation

Performance Analysis

Parameters for this scenario can be derived from known characteristics of vehicle traffic. For instance, on an autobahn there are 75,000-100,000 cars/d = 10,000 cars/h = 3 cars/s.

- Sending (normal state): Assumptions: 5 sensor information (rain, ice, temperature) from each car all 30 s in the normal state; but information are only relevant in a certain area with e. g. 10,000 vehicles -> 333 Updates/s
- Sending and receiving (emergency state): Sudden sending and receiving of information in emergency case to immediately affected vehicles (e. g., by an ad hoc network). Other vehicles can be informed with a certain delay.

Conclusion: This scenario integrates many different aspects of location-based systems and covers many components developed by different sub-projects within the center of excellence.

6.4 Security Considerations

Ensuring security and privacy may results is a significantly higher effort to realize services. If the location information of users must be protected against unauthorized access, this can require

- the exchange of additional messages (e. g., for key distribution),

- an increase in the amount of data to be transmitted (e. g., due to ciphering),
- more processing capacity (e. g., for ciphering, signature processing, verification), and/or
- architectural changes (e. g., because certain functions must not be performed within a certain context server).

This effort must be considered in addition to the functional requirements and may impose further scalability limitations.

An example for such implications is are privacy requirements for the Nexus location service and the event service (see [43]): If a user does not trust the location service and event service are, the user may, amonst others,

- store ciphered location information within the location service, or
- mix the correct location information with fake data in order to disguise the actual position.

In the former case, a centralized location service cannot evaluate events, i. e., the event service must be realized by the clients. This requires much more communication with the clients over typically bandwidth-limited access networks. In the latter case, much more events may be triggered, which have to be verified e. g. by the clients. Again, this requires much more signaling. In both cases, it is much more challenging to build a scalable system as if security and privacy issues were not considered.

7 Literature

[1] D. A. Menascé and V. A. F. Almeida, “Capacity Planning for Web Services”, Prentice Hall, Ed. 2, 2002

[2] ITU-T recommendation E.800

[3] F.-J. Banet, A. Gärtner, and G. Teßmar, „UMTS - Netztechnik, Dienstarchitektur, Evolution“, Hüthig Verlag, 2004

[4] L. Skorin-Kapov, D. Huljenic, D. Mikic, and D.Vilendecic, “Analysis of End-to-End QoS for Networked Virtual Reality Services in UMTS”, IEEE Communication Magazine, April 2004

[5] T. Buchholz, A. Küpper, and M. Schiffers, “Quality of Context: What It Is And Why We Need It”, 10th Workshop of the HP OpenView University Association (HPOVUA’03), Geneva, Switzerland, July 2003

[6] M. Scharf, “On the Response Time of Large-scale Composite Web Services”, Proc. of the 19th International Teletraffic Congress (ITC 19), Beijing, 2005

[7] B. Devlin, J. Gray, B. Laing, and G. Spix, “Scalability Terminology: Farms, Clones, Partitions, and Packs: RACS and RAPS”, Microsoft Research, MSR-TR-99-85, 1999

[8] V. Cardellini, E. Casalicchio, M. Colajanni, and Philip S. Yu, “The State of the Art in Locally Distributed Web-server Systems”, ACM Computing Surveys, vol. 34, no. 2, 2002, p. 263-311

[9] P. Jogalekar and M. Woodside, “Evaluating the Scalability of Distributed Systems”, IEEE Transactions in Parallel and Distributed Systems, Vol. 11, Nr. 6, 2000, pp. 589-603

[10] P. Jogalekar and C. Woodside, “A Scalability Metric for Distributed Computing Applications in Telecommunications”, Proc. Fifteenth International Teletraffic Congress (ITC-15), 1997, pp. 101-110

[11] André B. Bondi, “Characteristics of Scalability and Their Impact on Performance”, Proc. of the Second International Workshop on Software and Performance (WOSP 2000), Ottawa, Ontario, Canada, Sept. 2000, pp. 195-203

[12] J. Charzinski, J. Färber, and N. Vicari, „Verkehrsmessungen und Lastmodellierung im Internet“, PIK, Vol. 25, No. 2, April-June 2002, pp. 64-72

[13] B. Arbter, F. Bitzer, S. Bürklen, D. Dudkowski, M. Scharf, “Approaches for Modelling Mobility in Nexus”, SFB 627 Bericht Nr. 2006/09

[14] M. Wermuth, „Modellvorstellung zur Prognose“, in G. Steierwald, H.-D. Künne, (Hrsg): Straßenverkehrsplanung - Grundlagen - Methoden - Ziele, Springer Verlag, Berlin, 1994

[15] „Kraftfahrzeugverkehr in Deutschland“, http://www.verkehrsbefragung.de/page10.xml

[16] „Mobilität in Deutschland“, Studie des Bundesministeriums für Verkehr, Bau- und Wohnungswesen, http://www.kontiv2002.de

[17] “Dateline: Design and Application of a Travel Survey for European Long-distance Trips based on an International Network of Expertise”, http://www.socialdata.de/leistungen/dateline_d.php

[18] „Deutsches Mobilitätspanel“, http://mobilitaetspanel.ifv.uni-karlsruhe.de

[19] „Intermodale Vernetzung“, http://verkehrspanel.ifv.uni-karlsruhe.de

[20] A. K. Dey and G. D. Abowd, “Towards a Better Understanding of Context and Context-Awareness”, Workshop on “The What, Who, Where, When, and How of Context-Awareness”, Conference on Human Factors in Computing Systems, 2000

[21] D. Nicklas, M. Großmann, T. Schwarz, S. Volz, and B. Mitschang, “A Model-Based, Open Architecture for Mobile, Spatially Aware Applications”, Proc. 7th International Symposium on Spatial and Temporal Databases (SSTD 2001), Redondo Beach, California, USA, 2001

[22] “Google Earth” web site, http://earth.google.com

[23] U. Leonhardt, “Supporting Location-Awareness in Open Distributed Systems”, PhD. Thesis, Department of Computing, Imperial College, London, UK, 1998

[24] M. Bauer, F. Dürr, J. Geiger, M. Großmann, N. Hönle, J. Joswig, D. Nicklas, and T. Schwarz, “Information Management and Exchange in the Nexus Platform”, Version 2.0, Technischer Bericht, SFB Nexus, 2004

[25] P. Krishnan, D. Raz, and Y. Shavitt, “The Cache Location Problem”, IEEE/ACM Transactions on Networking, Vol. 8, Nr. 5, 2000, pp. 568-582

[26] Eric A. Brewer, “Lessons from Giant-Scale Services”, IEEE Internet Computing, 2001

[27] D. A. Menascé, V. A. F. Almeida, R. Riedi, F. Ribeiro, R. Fonseca, and W. Meira Jr., “A Hierarchical and Multiscale Approach to Analyze E-business Workloads”, Performance Evaluation, Vol. 54, Issue 1, 2003, pp. 33-57

[28] S. Rupp, „Kontextbasierende Systeme - Ansätze aus der Telekommunikation“, Industriekolloquium, Nexus SFB 627, Universität Stuttgart, 2005

[29] N. Nantavechsanti, “Simulations of Three Different Approaches for a GSM-Network Design”, Master Thesis, IKR, University of Stuttgart, 2004

[30] Microsoft Map-Point, http://www.microsoft.com/mappoint/

[31] IntelliWhere Location Server, http://www.intelliwhere.com

[32] A. Leonhardi, „Architektur eines verteilten skalierbaren Lokationsdienstes“, Dissertation, IPVS, Universität Stuttgart, 2003

[33] T. Drosdol, T. Schwarz, M. Bauer, M. Großmann, N. Hönle, D. Nicklas, “Keeping Track of "Flying Elephants": Challenges in Large-Scale Management of Complex Mobile Objects”, Proc. of INFORMATIK 2004, Ulm, Germany, 2004

[34] D. A. Menascé, “Response-time Analysis of Composite Web Services”, IEEE Internet Computing, vol. 8, no. 1, pp. 90-92, 2004

[35] R. Nelson and A. N. Tantawi, “Approximate Analysis of Fork/join Synchronization in Parallel Queues”, IEEE Trans. on Computers, vol. 37, no. 6, pp. 739-743, 1988

[36] D. A. Menascé, D. Saha, S. C. da Silva Porto, V. A. F. Almeida, and S. K. Tripathi, “Static and Dynamic Processor Scheduling Disciplines in Heterogeneous Parallel Architectures”, Journ. of Parallel and Distr. Comp., vol. 28, pp. 1-18, 1995.

[37] B. Cahoon, K. S. McKinley, and Z. Lu, “Evaluating the Performance of Distributed Architectures for Information Retrieval using a Variety of Workloads”, ACM Trans. Inf. Syst., vol. 18, no. 1, pp. 1-43, 2000

[38] M. E. Crovella and A. Bestavros, “Self-similarity in World Wide Web Traffic: Evidence and Possible Causes”, IEEE/ACM Trans. Netw., vol. 5, no. 6, pp. 835-846, 1997

[39] U. Vallamsetty, K. Kant, and P. Mohapatra, “Characterization of E-commerce Traffic”, Electronic Commerce Research, vol. 3, no. 1-2, pp. 167-192, 2003

[40] Deutscher Blinden- und Sehbehindertenverband e.V.

[41] T. Schwarz, M. Iofcea, M. Grossmann, N. Hönle, D. Nicklas, B. Mitschang, “On Efficiently Processing Nearest Neighbor Queries in a Loosely Coupled Set of Data Sources”, Proc. 12th ACM International Symposium on Advances in Geographic Information System (ACM GIS 2004), Washington D.C., 2004

[42] M. Bauer, L. Jendoubi, O. Siemoneit, “Smart Factory - Mobile Computing in Production Environments”, Proc. of MobiSys 2004 Workshop on Applications of Mobile Embedded Systems (WAMES 2004)

[43] A. Gutscher, “Coordinate Transformation - A Solution for the Privacy Problem of Location Based Services”, Proc. of the 2nd International Workshop on Security in Systems and Networks (in conjunction with IEEE International Parallel and Distributed Processing Symposium), 2006