# The World-Wide Web of the Future
# A New Generation of a Distributed OS

Peter Kutschera and Friedemann Schwenkreis
IPVR, University of Stuttgart
Breitwiesenstr. 20-22
D-70565 Stuttgart, Germany
email: {kutscher,schwenk}@informatik.uni-stuttgart.de
URL: http://www.informatik.uni-stuttgart.de/ipvr/as/personen/{kutschera,schwenk}.html

## 1 Introduction

The evolution in technology during the last few years has led not only to an increase in processor speed of about 200 percent every year, but also to an improvement in the areas of tertiary storage techniques and communication networks that are now capable to deliver data at a gigabit rate. Parallel to the performance increase, the prices for hardware decreased and made it possible to realize an increasing number of new applications. Especially in the area of multimedia computing, developers began to realize a lot of applications ranging from video-on-demand to computer based learning and systems for cooperative work. These applications all had in common that they were characterized by their developers as the killer application of the forthcoming generation of multimedia computing.
At the same time, it became obvious that these kind of services are wanted to be accessible from peoples' home. Hence, high-speed networks to deliver audio- and video data in real-time must be available. Since the infrastructure at that point in time was not powerful enough to fullfil the requirements, the governments of different countries started several activities to eliminate the shortcomings. The most popular activity is the "National Information Infrastructure" (NII) initiative which has been established in the United States. Its goal is to interconnect government institutions, schools, universities, public institutions and companies with high-speed networks, the so-called „Information Highways".
A few years before started a project at the European Center for Nuclear Research (CERN) in Geneva which has almost five years after its start become the most popular multimedia application and represents the real killer application for high-speed networks. The goal of the World-Wide Web (WWW) project, was the implementation of a distributed hypermedia system. The idea behind hypermedia systems is the management of non-linear information consisting of text, images, video and audio data, which can be linked to each other. These links enable the user to navigate through the information. The research at CERN was not noticed from a broader audience until the National Center for Supercomputing Applications (NCSA) developed the graphical user interface Mosaic which is nowadays a well-known name. The first version of the browser was developed in 1993 and made the World-Wide Web for Unix workstations (connected to the Internet) accessible. From now on the World-Wide Web gained more and more popularity and in 1994 there were already 1500 World-Wide Web servers in operation. In 1995 the World-Wide Web made its way into mass media and introduced an increasing number of commercial on-line services. Currently, there are more than 7000 World-Wide Web servers accessible.
The availability of the WWW software (servers and browsers) on almost every hardware platform as well as the possibility to access world-wide resources through an easy-to-use graphical interface makes the web an ideal basis for both commercial and non-commercial services.

## 2 Shortcomings of the web

It is commonly agreed that the popularity of the web will still rise and the development of languages such as Java, which marks a shift in the programming paradigm from desktop to the net, as well as the announcement of computer companies to ship Internet-PCs, support that point of view. If cable companies come up with reasonable prices and a secure way to transfer money is available, we are soon facing the situation of millions of web users and millions of service providers on the web. But the only way this vision comes to reality is the elimination of today's shortcomings which will be shortly discussed in the following.

### 2.1 Finding services.

Nowadays with only some thousands of services on the web, there are two possibilities for the user to find a specific service. The first method is to try a good guess about the address of the companies web server (choose www as a prefix, followed by the name of the company and add the postfix com to the name). This method presumes that you already know which company provides the wanted service. If one just knows the service this method cannot be used and the only way to find your needs is to use a directory service such as Yahoo or start a keyword search using a web robot like the World-Wide Web Worm [1] or WebCrawler [3]. In an environment with millions of services offered within the web, the second method will become far more important. Today's methods cannot be used, because an on-line search will last for days, you have to pay for it, and central directories will become a bottleneck. Such a well-known service like (regional) yellow pages is currently not available on the web.

### 2.2 Support for the definition of complex activities.

A standard user activity incorporates the invocation of different services based on the results of previous interactions. For example, a user requests a specific service and based on the outcome of this service further service requests are initiated. Nowadays, the user has to keep track of his activity by himself without any system support. This is clearly not what the user wants, he or she just wants to specify a complex activity containing several sub-activities on a higher level of abstraction. The system should be responsible for the coordination and the invocation of the sub-activities. At the end of the whole activity, the user gets the result showing the outcome of each sub-activity. If the complex activity could not be fulfilled, the user wants to receive an appropriate message and all of the changes that took place during the execution of this activity have to be backed out.

### 2.3 Failure handling.

If the user nowadays tries to access services through the World-Wide Web, he or she gets simply an error message that his or her request could not be processed. However, no detailed information about the occurred error is provided to the user. For instance, transient and permanent errors cannot be distinguished. Another problem is inherent in the structure of the web where the server acts just as a frontend for a real world application on the backend. The user normally gets a message when his or her request has been processed at the frontend, but if something goes wrong at the backend, no information about that specific error is available. Furthermore, the user has no possibility to get the status of a request, e.g. after an unexpected long period of execution time.

## 3 Enhancing the web

Since web browsers become more and more the general purpose tool for end users, the web itself becomes more and more a high-level distributed operating system platform. Furthermore, with the availability of "net-aware" programming languages like Java it will be a very powerful platform for application development. However, the complexity of such a world-wide operating system is enormous and causes new problems (see above). The following sections will present some approaches which will help to solve some of the problems. Due to limited space we will just give a survey of the ideas without going into much detail.

### 3.1 Traders and mediators

The huge number of services on the web requires the introduction of special "meta services" to support users to find specific services which solve their problems. These so-called traders enhance the net by adding semantic information to simple addresses and supporting context dependent search request, i.e. a trader keeps track of what a customer already knows. Furthermore, traders provide information about the quality of the services they know. They collect statistical information of network performance and response times to mediate optimal services. More or less, they can be seen as a part of the load balancing system of the web.
Search engines like Yahoo are first steps into the direction of traders. Their capabilities are still very limited

but they show how information can be provided to ease the search on the web. Additionally commercial service providers come up with such supporting servers and users are willing to pay for that - it saves cost and time to use such a service instead of searching on your own.

## 3.2 User agents

Searching on the web is a good example for a major problem of the web. Every service request has to be done in a synchronous way, i.e. the user has to have a connection to the server until the results of a request are received. Since the majority of users is charged for the connection time, this way of operation is unsatisfactory. Due to increasing processing times in case of the search (see section 2) without predictable results, it is possible that one has to pay a relatively high amount of money for a zero result.
The main reason for this drawback is that servers are in general state-less. Hence, results of service requests have to be passed directly to the user's representing agent which is nowadays just the web browser. The solution to the problem proposed by us is the introduction of more powerful user representing agents (e.g. on the basis of Java) which support an asynchronous mode of operation.
The idea is, that users do not just issue a service request. Instead, they send an agent [2] to the server which is the mediator between the server and the user's web browser. The agent is responsible for sending the service request to the server (local communication), collecting the results and exchanging information with the user's web browser (on demand).
The major advantages of this approach are obvious. Connection times can be reduced if the time needed for processing the request is greater than the time needed to transfer the agent. Additionally, status requests as well as the transfer of results can be made asynchronously using the agent (which also supports mobile users). However, service providers have to offer appropriate resources to enable users to send their agents to the services.

## 3.3 Processes of the web - workflows

The simple agents of the previous section are still not powerful enough to satisfy the needs. Especially, in case of applications in which multiple services have to be accessed, a much more powerful paradigm has to be introduced: *workflows*. To define web based applications a mechanism is needed which allows to access several services in an arbitrary sequence. Taking into account that users who initiated such an application do not want to be connected until the execution is finished, we have to extend the notion of the agents presented in the previous section.
First, user agents have to become mobile, i.e. an agent must have the capability to migrate from one service location to another. Furthermore, to support parallel execution an agent must be able to *fork* other agents which are responsible for the parallel branches. The parent agent is still responsible for the collection of the results and for the synchronization of the different branches.
Several challenges come with that flexible mode of operation. The web as the platform of agents has to support the migration of the agents. Additionally, users as well as agents must be able to contact other agents without knowing their physical location. Since users want to get information about the progress of their agents, there must also be an interface for status requests. However, it would be a bad idea to collect all of the status information just on demand. Instead, we propose that the first agent representing the user on the web is collecting the status information of all "sub-agents". This means that sub-agents have to send updates on their status to their "super-agents" when a significant state change has happened.

## 3.4 Failure semantics and recovery

It is our observation that processes on the web will run for a relatively long duration (ranging from minutes to days and more). Therefore and due to the asynchronous mode of operation, defined failure semantics and recovery issues become highly important. Especially, the agents representing a user do have to guarantee their availability as well as a consistent view on the current state of the task for which they are responsible.
To be more specific, for users it is not acceptable that work already done on behalf of a request gets lost if the machine on which their agent is located goes down. Hence, at least the user representing agents must be recoverable, i.e. after the machine comes up again the agents must be recovered and set to a state in

which they are able to continue their work. But this forward recoverability can only be realized if the sub-agents do also have defined failure semantics.
In case of sub-agents the forward recoverability is not always necessary. If sub-agents do have for example transactional properties (ACID), which results in a guaranteed backward recovery after a failure, the user's agent would be also able to execute its forward recovery (this is similar to the ConTract approach [4]). However, if a sub-agent is just able to guarantee forward recovery, each of its super-agents must be forward recoverable too.

# 4 Summary and conclusions

In this paper we have presented our view on the future of the world-wide web. Since several requirements correspond to nowadays operating system features, we think that web of the future can be seen as a high-level distributed operating system platform. Several problems have been presented and some of them can also be found in today's operating systems. The presented approaches to solve these problems have also (at least partially) been proposed for classical operating systems. For instance, transactional features and defined failure semantics can be found in some OSs.
The mobile agent approach provides a highly flexible mode of operation introducing new classes of problems. Due to limited space, we have not discussed for instance accounting or security problems. The accessibility of an agent in a world-wide(!) environment is also not a trivial problem. However, we do believe that today's service providers which "sell" the access to the net must enhance their services by the presented features to be able to handle millions of users.
Since web browsers are becoming the general purpose tools of users, there will be more and more the requirement that applications can be run inside of the browser. The "Java revolution" shows that this vision becomes true. Even though, Java in its current state is just a starting point. Our view of agent based computing on the web can be realized easily as an extension of Java. Furthermore, since applications written in Java can be controlled very well (they are running in a controlled address space), the features to guarantee the failure semantics can be integrated in the underlying platform.

# References


[1] O. A. McBryan, GENVL and WWWW: Tools for taming the web, Proceedings of the First International World-Wide Web Conference, 1994

[2] B. Knaster, Presenting MAGIC CAP, Addison-Wesley, 1994

[3] B. Pinkerton, Finding what people want: Experiences with the webcrawler, Proceedings of the Second World-Wide Web Conference, 1994

[4] A. Reuter, F. Schwenkreis, ConTracts - A Low-Level Mechanism for Building General-Purpose Workflow Management Systems, Bulletin of the Technical Committee on Data Engineering (IEEE Computer Society), Vol. 18, No. 1, 1995