Real-Time Communication in Wireless Home Networks

Hans Scholten and Pierre Jansen
University of Twente
Department of Computer Science (EEMCS)
P.O.B. 217, 7500 AE Enschede, the Netherlands
email: {scholten,jansen}@cs.utwente.nl

Abstract

This paper describes a medium access protocol for real-time communication in wireless networks. Medium access is controlled by a scheduler, which utilizes a pre-emptive earliest deadline first (PEDF) scheduling algorithm. The scheduler prevents collisions in the network, where normally only collisions are avoided. PEDF is the scheduler of choice, because it has excellent properties with respect to bandwidth utilization, dynamic behaviour and feasibily analysis. The scheduler can be deployed in managed networks, where it resides in the base station, as well as in peer-to-peer or ad hoc networks, were it is distributed over the stations. The protocol is simulated and an implementation based on IEEE 802.11b is realized.

Index Terms

Real-time communication, medium access scheduling, wireless network, earliest deadline first

I. Home Networking

Since some time now networks are appearing in the private environments of users. In most cases the reason to have such a network is to interconnect PCs, and to connect these PC's to a single gateway to the Internet. Now applications start to appear that use this private network for other purposes as well. An example is recording live video on a server and watching these recordings anywhere at home on any available PC. However, there is other connectivity as well in the average home, e.g. a telephone network, a cable TV network, a 'network' that connects the video recorder to the TV, a thermostat connected to the central heating system, sensors at doors and windows connected to the burglar alarm. It is therefore that the next generation of domestic networks sees the challenge of integrating all these different nets into one coherent network.

Domestic integrated networks must support different classes of appliances and applications, each with its own characteristics:

- *entertainment class:* the network traffic consists mainly of isochronous streaming media, like audio and video. This class requires a high bandwidth and real-time responses;
- *command and control class:* the network contains sensors and actuators, like fire detection, central heating control and burglar alarm. These devices use or generate low bandwidth data. Often these command and control devices need real-time services and these services must be dependable;
- *information class:* this class consists of the traditional PC and Internet applications. Network traffic is bursty in nature and only needs best effort responses.

Domestic networks must face other challenges as well. These concern:

- *compatibility and interoperability*: all appliances in the home will be connected and have to work together. So they must 'speak' the same (sub-)set of protocols;

This work is sponsored by the Netherlands Organization for Scientific Research (NWO) under grant number 612.060.111, and by the IBM Equinox programme

- *data compatibility*: the appliances must all be able to work on the data in the network. E.g. it should not matter on which device someone listens to pre-recorded music. So data formats for different devices must be compatible or must be converted on-the-fly;
- *resources*: the network will contain both resource-lean and resource-rich devices (in terms of CPU, memory, power, size). But all must be connected to the network and work together, regardless of resources;
- *ad hoc configuration and service discovery*: a domestic network is dynamic. Devices are connected to the system and removed again. It would be undesirable to configure the system by hand every time this happens, the system must configure itself automatically.

HAVi [1], Jini [2] and Universal Plug-and-Play [3] are just a few examples of initiatives for domestic networks. They use different types of network, wireless and wired, but often lack guaranteed real-time behaviour, require substantial resources, or are suitable only for one class of applications. In general, this makes these systems unsuitable for *integrated* networks for domestic use.

The remainder of this paper focuses on real-time behaviour in wireless home networks. This is an important aspect of such networks, because it is not only needed to stream multimedia, but also to ensure command and control data transfers in a reliable way. Real-time behaviour is obtained by the deployment of scheduling techniques from the operating systems world. The paper describes a real-time protocol that is suitable for single-hop networks and, with restrictions, for multi-hop networks as well.

II. Token Scheduling

Scheduling techniques for real-time communications in networks is not new. IEEE802.4 token bus, IEEE802.5 token ring and FDDI are examples of networks that achieve real-time behaviour. The main properties are described by Malcom and Zhao [4] and by Sevcik and Johnson [5]. Most real-time systems pass a token around and during one rotation of the token all nodes in the network are visited. In general some bandwidth is wasted, because (1) all nodes are visited, instead of only those nodes that use the network, and (2) the schedule is not flexible and dynamic enough to utilize all available bandwidth.

The protocol we propose also uses a token, however it does not follow a round robin or similar schedule. Instead the token is scheduled to visit only those nodes that need servicing. The token is not simply passed on to the node next to the node holding the token in a round robin fashion, but the token is passed on, based on a pre-emptive earliest deadline first (PEDF) schedule. This scheme is applicable to both peer-to-peer and managed networks.

The scheduler assumes periodical streams, where each stream may have its own period. During one period a frame or a set of frames is transported. The start of the period is the earliest possible time a frame becomes available and the end of the period is the latest possible time a frame must have been sent. The end of a period is equal to the start of the following period of the same stream. The period can be seen as a time window in which a node can send a frame. The fraction of time needed to send the frame over the total time of the period is the bandwidth of the stream. Thus a stream can be characterized by a period and a bandwidth which is also a natural way to express the parameters of the token scheduler.

In the peer-to-peer configuration no master that calculates a schedule and distributes the token is present in the system. Instead, the scheduler is present in every node in the network. The calculated schedule piggy-backs the token. So, when a node receives the token, it also receives the most recent schedule for bandwidth partitioning. If the node wants to negotiate another period or bandwidth for its stream(s), it can calculate a new schedule and act on this new schedule. If the node just wants to send its stream according to the previously agreed on schedule, it will do so.

The same mechanism is valid in a managed network, where a base station divides bandwidth over the participating stations. Now all scheduling decisions are made in the base station and stations can request bandwidth. Eighty percent of the bandwidth is reserved for real-time communication. During that time only those nodes that are included in the schedule can use the network in turn, because they will receive the token once in a while. To allow newly arriving nodes in the network, or nodes that do not

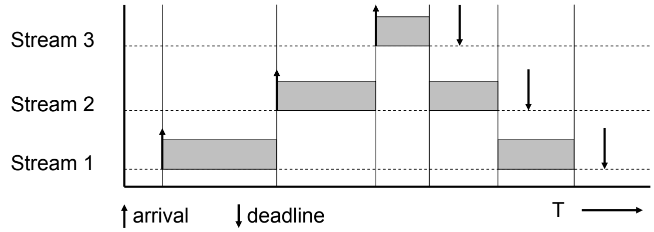

Fig. 1. Pre-emptive EDF scheduling

communicate yet, to add streams, the remaining twenty percent of the bandwidth is used to send the token around, visiting all nodes in the network. Considerate efforts have been made to prevent token loss or token duplication. A more detailed explanation can be found in [6] or [7].

A. The Scheduler

In the following we assume a peer-to-peer or ad hoc network, but the principles are equally applicable to managed networks.

The distributed scheduler resides in every network node, but one might say that the scheduler travels with the token, as the token contains the up-to-date information for the scheduler and at any time there is only one node that schedules the network, which is the node with the token. The scheduler is a pre-emptive earliest deadline first scheduler. This type of scheduler schedules the stream that has the earliest deadline first. It is pre-emptive because an instance of a periodical stream can be interrupted by another stream if this stream has an earlier deadline, but arrives later. Arrival time of a stream in this context is the time when the data becomes available to be sent by the network.

An example of a schedule of a set of three streams, each with an arrival time (↑) and a deadline (↓), is shown in figure 1. Stream 1 arrives first and starts sending. Stream 2 arrives at a later time, but it has an earlier deadline, so stream 1 is pre-empted by stream 2. In its turn stream 2 is pre-empted by stream 3. After stream 3 has finished sending its data in the current period, stream 2 will resume sending. And in the same fashion stream 1 resumes sending when stream 2 has finished.

The pre-emptive earliest deadline first algorithm is chosen because it has some properties which are appropriate for network scheduling. The first and most important one is that it has a utilization factor of one hundred percent. Thus available bandwidth in the network is used in an extremely efficient way. The second advantage is that schedules are dynamic and can be calculated in real-time. This allows streams to be added to and removed from the network dynamically. Schedules are calculated on-line and on-the-fly. The third advantage is the fairly simple feasibility analysis of a schedule. Feasibility analysis determines whether a new stream can be added to the active set of streams or not, or whether an existing stream can change its period or requested bandwidth. A disadvantage is that PEDF scheduling performs unpredictable in the presence of an overload. When the set of active streams requires more bandwidth than is available, it is not certain which stream will ”run”. However, because overloads are avoided by the feasibility analyses this is a situation which will not occur under normal circumstances. For more information and formal proofs on PEDF see [8].

B. Feasibility Analysis

A main concern in a real-time system is that a (computational) task may never miss its deadline. Before a set of tasks is accepted and can execute, the scheduler must verify that the task set is feasible and will never cause a task to miss its deadline. When the task set changes, because a new task is added or the characteristics of a task change, a feasibility analysis must be performed on the new task set. When the feasibility analysis shows that the new task set can not be scheduled, it must be rejected. In the following we assume that for a periodic task the start of a period is equal to the deadline of the previous period. This is a common requirement when a task processes multimedia data streams. E.g. a task (or process) that sends a video stream must send each video frame before the next one becomes available, or else the stream will hiccup and possibly an overrun error will occur. Under this assumption a set of tasks is schedulable with the pre-emptive earliest deadline first algorithm if and only if

$$\sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1$$

Where:
n: number of tasks,
C_i: Computation time of task i,
T_i: Period of task i.

This is true for a set of computational tasks, but can be applied to network scheduling as well [6].

The fraction of bandwidth that is requested by stream i is equal to $\frac{B_i}{B}$. The total of bandwidth fractions can not exceed 100 percent of the total network bandwidth. Thus, a set of streams is schedulable with PEDF if and only if

$$\sum_{i=1}^{n} \frac{B_i}{B} \leq 1$$

Where
n: number of streams,
B_i: Bandwidth of stream i,
B: Maximum bandwidth of the network.

When the streams in the network meet this requirement, the PEDF scheduler will find a schedule.

C. Monitoring

One of the most hazardous things for a token-based network is the loss of the token. This could happen due to an error in the transmission of the token or due to a crash of the node holding the token. To solve most of these problems, the network uses monitors. Every node in the network that participates in the real-time phase can become monitor. When the active node relinquishes the token, it becomes the monitor for the next active node and will observe its behaviour. The first action of the monitor is to wait for the acknowledgement that the new active node has received the token in good order. When the monitor does not receive that message, it generates a new token. As the monitor was the last node to hold the token, a new token can easily be generated. To make sure it has not crashed the active node will send another acknowledgement at the end of its active state. Because the monitor knows the schedule, it knows when to expect this acknowledgement. When this time, plus some extra time, has passed without receiving the acknowledgement, the monitor assumes the node has crashed and removes it from the network and re-schedules the network.

The following list summarizes detectable cases of token loss:

- token does not arrive at new node;
- reply from transmitter is lost;
- transmitter dies before sending a reply;
- monitor dies.

There are some token loss situations that cannot be detected quickly. No node will receive a token anymore and all will timeout and go offline. It will take a few seconds until they reinitialize the network and generate a new token. Basically the whole network is reset and all current real-time streams and node information is deleted. This has a huge negative impact on the network applications because they all lost their deadlines. The following situations cause a network reset:

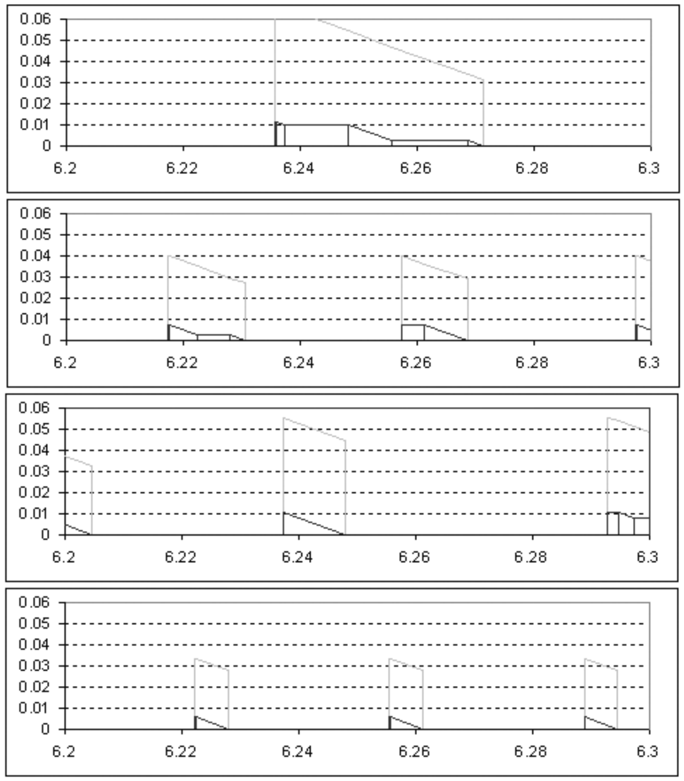

Fig. 2. PEDF schedule of a set of periodic streams

- transmitter dies after sending a reply;
- both nodes die simultaneously.

Normally the transmitter tells the monitor that it is alive but what if the poll or reply also becomes garbled? The monitor thinks that the transmitter is dead and it will generate a new token. Now two tokens are present in the network, one at the monitor and one at the transmitter. There is no foolproof method to detect token duplication but there are some hints the network will give when the token is duplicated:

- a token arrives at a node that already holds a token;
- a stream misses its deadline because it cannot resolve its bandwidth requirements.

In this case recovery of the system depends on the particular situation, but probably results in a network reset.

D. Simulation and Prototype

The network and its PEDF token mechanism are simulated and a prototype based on IEEE 802.11b is built. Figure 2 shows some simulator output for the PEDF scheduling of a set of periodic streams. Only part of the streams from this set is shown. The grey (light) line depicts the time left until the next absolute deadline for that stream. This time decreases linearly and will cross the horizontal axis at the start of the next period for that stream. The period of a stream equals the time between two deadlines. The black line is the remaining streaming time for that stream during that period. When the stream is transmitted this line decreases linearly with the number of bytes sent. This line must reach zero before the start of the next period. A horizontal black line indicates that no data is transmitted. Normally this means that the stream is pre-empted by another stream with an earlier deadline.

PEDF ensures that the total network utilization is 100 percent. However, some overhead must be taken into account. Some figures calculated from the simulation are: average overhead per stream related to total bandwidth: 0.54 percent; average worst-case overhead per stream related to total bandwidth: 0.78 percent;

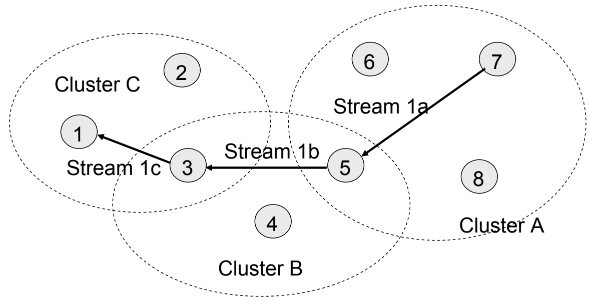

Fig. 3. Example of a multi-hop network

average overhead per stream related to effective stream utilization: 1.33 percent; and average worst-case overhead per stream related to effective stream utilization: 1.95 percent. An overview can be found in [9].

III. Multi-Hop Networks

The wireless prototype is a single-hop network and uses the ad hoc mode. Care is taken that the network contains no hidden nodes. Measurements of sending and receiving streams, pre-emption of streams and feasibily analysis are in line with the theoretical predictions. See [10].

We verified that a real-time network based on pre-emptive earliest deadline first token scheduling is feasible, and that this principle is applicable to single-hop wireless networks. However, in a wireless ad hoc network hidden nodes can not always be avoided. This invalidates the assumptions made so far and adaptations to the protocol are needed. These are related to the following issues:

- route discovery;
- data forwarding;
- token passing.

In the following we address some of these issues in multi-hop networks.

Requirements on ad-hoc networks are virtually impossible to hold in real-time networks. For example, mobility of nodes introduces a nondeterministic network topology, making it impossible to guarantee routes to be valid for a long enough time. However, in home network environments, assumptions can be made on node availability, mobility, and distance. These assumptions loosen the requirements a lot, allowing new strategies to be developed:

- Nodes are mobile, however, their movement is limited. Most devices have a more or less fixed position in the house.
- Distance between nodes is limited. The signal range of a node is not much less than the size of the house. As a result, the diameter of the network will be small, but may be greater than one.
- At start-up, a fully available network with working connections between any two nodes, is not yet required. A few seconds time is allowed between switching a device on and actual data transmission.
- Some rerouting time after the occurrence of a broken link is allowed.
- The number of nodes in the network is finite and not very large: a few dozens at most.
- The token visits each node frequently. Nodes do not hold the token for a very long time.

The consequence of these assumptions is that routes will be discovered within reasonable (bounded) time, and that discovered routes are likely to last a fairly long time. So all nodes are reachable, routes to nodes are known, and the number of hops to reach a node is known.

In the example of figure 3 a stream from node 7 to node 1 is relayed by intermediate nodes. The network is divided in clusters, where every cluster is a fully connected (sub)network. Some nodes are in more than one cluster, which makes them potential relays for transmissions from one cluster to an other. This means that the stream is divided in one sub-stream per hop, and that each frame of the stream will be transmitted one after the other. This is shown in figure 4, where one period of the stream is shown

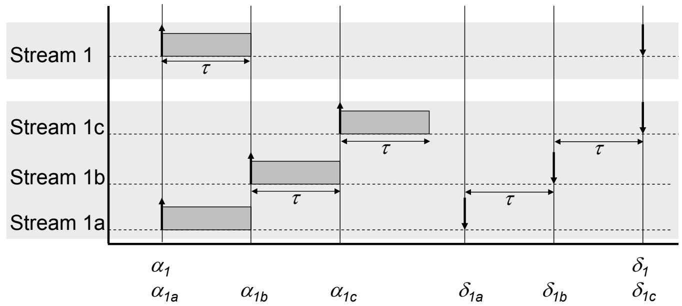

Fig. 4. Earliest multi-hop PEDF schedule

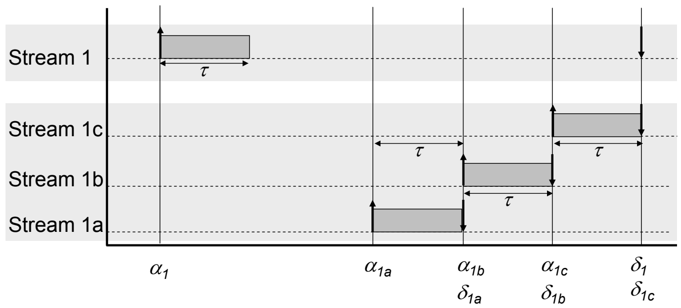

Fig. 5. Latest multi-hop PEDF schedule

with an arrival time α_1 and a deadline δ_1. Sub-stream 1_a has the same arrival time α_{1a} as stream 1, and sub-stream 1_c has the same deadline δ_{1c} as stream 1. Because stream 1_b has to be received by node 3 before stream 1_c can be sent by the same node, the deadline δ_{1b} of stream 1_b equals the latest possible arrival time α_{1c} of stream 1_c:

$$\begin{array}{lr} \alpha_1; \delta_1 & (stream 1) \\ \alpha_{1a} \geq \alpha_1; \delta_{1a} \leq \delta_{1b} - \tau & (stream 1_a) \\ \alpha_{1b} \geq \alpha_{1a} + \tau; \delta_{1b} \leq \delta_{1c} - \tau & (stream 1_b) \\ \alpha_{1c} \geq \alpha_{1b} + \tau; \delta_{1c} \leq \delta_1 & (stream 1_c) \end{array}$$

Figure 4 shows a possible scheduling of the sub-streams. Here all streams are scheduled as early as possible. Figure 5 also shows an alternative scheduling of the sub-streams, but now all streams are scheduled as late as possible. The scheduler has to ensure that stream 1_a is transmitted before stream 1_b, and stream 1_b before stream 1_c. This is guaranteed by the PEDF scheduler when:

$$\begin{array}{l} \alpha_{1a} \leq \alpha_{1b} \leq \alpha_{1c} \\ \delta_{1a} < \delta_{1b} < \delta_{1c} \end{array}$$

The limits for α's and δ's for the sub-streams meet these requirements.

Now a set of streams, where each stream i is composed of s_i sub-streams, is feasible if and only if

$$\sum_{i=1}^{n} \frac{s_i \cdot B_i}{B} \leq 1$$

Where
n: number of streams,
s_i: number of sub-streams in stream i,
B_i: bandwidth of original stream i,
B: Maximum bandwidth of the network.

For reasons of clarity, overhead caused by extra token sending etc. is not taken into account. This –multi-hop– token scheme is a general model and is applicable to both ad hoc and managed network models: (1) for the managed network model with an access point that relays messages from one station to another, the number of hops, and the number of sub-streams, is two; (2) for the ad hoc fully connected network model the number of hops is always one; (3) and for the ad hoc with hidden nodes network model the number of hops is variable, but probably three or less, because of the relatively small size of a personal area network.

IV. Conclusion

In this paper we have described a protocol that is suitable for real-time communications in a wireless home network. The protocol is based on a pre-emptive earliest deadline first scheduler. PEDF ensures a very efficient use of available bandwidth, yet its feasibility analysis is simple enough to be implemented in small embedded systems. A prototype that uses the protocol is realized on a wireless IEEE802.11 network. Measurements in this prototype correspond with the results obtained by the simulations. The prototype assumes a single-hop fully connected topology without hidden nodes, which, in ad hoc networks, is normally not guaranteed. Using the simple, fully connected topology as a starting point, a scheme is presented for real-time communications via relaying nodes or base stations. The feasibility analysis of the multi-hop scheme is an extension of the analysis of single-hop networks.

During the development of the multi-hop protocol, minor attention has been paid to behaviour as a result of topology changes, especially the invalidation of connections between nodes. If a topology change leads to infeasibility, a policy for dropping and/or suspending streams will have to be developed. In the multi-hop protocol, the infeasibility may be momentarily, since a new route may show up as soon as the updated route information has emerged through the network. Considerable improvements may be achieved by the use of multi-path. However, this does not instantly solve the instability caused by massive topological changes.

The assumptions formulated for home networks allow rerouting time after a topological change. Suspending the stream in the meantime, or skipping parts of it, may be a more attractive alternative to dropping a running data stream. Currently the multi-hop protocol does not support the storage of multiple routes to the same destination. For practical applications it is desirable not to be dependent of the availability of all nodes and connections in a single route. A more robust alternative is achieved by extending the protocol to store alternative routes to the same destination. Consequently the network is prepared for topological changes. Rerouting is not necessary and streams may continue without manual intervention.

The token-based approach guarantees that only one node in the network will transmit. However, for an ad-hoc network consisting of multiple, partly overlapping, clusters, this is more restrictive than necessary: only one node in a cluster may transmit. Depending on the particular configuration of the network, some parallelism may exist between clusters. If the real-time protocol is to be used in larger buildings, with a possibly large network diameter, it might be worth to investigate the possibilities to have multiple tokens in the network, each belonging to a different cluster. Cluster overlap will have to be taken into account to prevent collisions. An aid in designing a protocol that supports multi-cluster networks may be the use of a clustering protocol, which elects cluster heads from the nodes in a cluster. This node may act as a central unit that controls the traffic in its cluster.

References

[1] HAVi, "Home audio and video interoperability web site," Website: 'http://www.havi.org'.

[2] JINI, "Jini homepage," Website: 'http://www.jini.org' or 'http://www.sun.com/jini'.

[3] UPnP, "Upnp homepage," Website: 'http://www.upnp.org'.

[4] N. Malcom and W. Zhao, "The timed token protocol for real-time communications," *IEEE Computers*, vol. 10, no. 1, pp. 35–41, Jan. 1994.

[5] K. Sevcik and M. Johnson, "Cycle time properties of the fddi token ring protocol," *IEEE Transactions on Software Engineering*, vol. 13, no. 3, pp. 376–385, Mar. 1987.

[6] T. Hattink, "HOTnet, a real-time network protocol," Master's thesis, University of Twente, August 2001.

[7] J. H. Wijnberg, "Prototyping the real-time 'HOTnet' network protocol on ethernet," Master's thesis, University of Twente, May 2002.

[8] G. C. Buttazzo, *Hard Real-Time Computing Systems – Predictable Scheduling Algorithms and Applications*. Kluwer Academic Publishers, 1997, iSBN 0-7923-9994-3.

[9] J. Scholten, P. G. Jansen, F. T. Y. Hanssen, P. H. Hartel, T. Hattink, and V. Sundramoorthy, "Multimedia QoS in low-cost home networks," in *27th Annual IEEE Conference on Local Computer Networks (LCN)*. Tampa, Florida: IEEE Computer Society Press, Los Alamitos, California, Nov 2002, pp. 327–328.

[10] W. H. Mank and A. Zwikker, "Designing and prototyping the real-time rtnet protocol on wireless lan," Master's thesis, University of Twente, November 2002.