Technical Report

C-RATH
A Collaborative Adaptive Tutoring Hypertext System *

Cord Hockemeyer[†]

August 17, 2000

Department of Psychology
University of Graz, Austria
E–mail: Cord.Hockemeyer@kfunigraz.ac.at

*This work was financed by the European Commission through a Marie Curie Fellowship (Grant ERBFMBICT983377) entitled *A System for Developing Communication Spaces for a WWW Learning Environment* within the program "Training and Mobility of Researchers".
[†]In co-operation with Dietrich Albert (E–mail: Dietrich.Albert@kfunigraz.ac.at).

The software system described in this report can be accessed via the World Wide Web at the address http://wundt.kfunigraz.ac.at/crath/ . For questions about C-RATH, send an email to crath@wundt.kfunigraz.ac.at.
An online version of this report is available through the URL http://wundt.kfunigraz.ac.at/hockemeyer/publications.html .

Contents

1. Introduction

As Dillenbourg (1999) states that a definition for the term *Computer Supported Collaborative Learning* (CSCL) is very dificult to agree upon. He distinguishes between CSCL in a wider sense denoting any type of collaboration support and CSCL in a more narrow sense denoting collaboration support in problem solving (often based on constructivist learning theories). The work reported in this document belongs to the former type of CSCL, more concrete to the class of *Computer Mediated Communication* (CMC) systems (see, e. g. Veerman, Andriessen, and Kanselaar, 1999, or Ottmann and Tomek, 1998).

Current CMC systems in general simply offer electronic opportunities for group communication in form of , e. g., newsgroups, e–mail conferencing systems, or chat systems. Two approaches going beyond this pure technical support are the iDCLE system (intelligent distributed collaborative learning environment) developed by Inaba and Okamoto (1995, 1997) and Hoppe's (1995) *multiple student modeling*. The special merits of Hoppe's approach is the formal specification of the system. However, he focuses on communications in groups with a leader person (e. g. a teacher or a leading peer). Inaba and Okamoto investigate communication in "real" peer groups but they lack an underlying formal model. Their concept of different kinds of relations between pupils' utterances, however, suggests using a relational model for the communication. This is supported by the fact that using a relational model has already proven to be very useful in the construction of the adaptive hypermedia tutoring system `RATH` (Hockemeyer, Held, and Albert, 1998; Albert and Hockemeyer, 1997).

In the sequel, the `RATH` system and the underlying theories are described. Afterwards the models of Hoppe and of Inaba and Okamoto will be summarized. Chapter 2 then develops a formal foundation for a CMC system integrating the new ideas of Hoppe as well as of Inaba and Okamoto, and the concepts developed for and applied in the `RATH` system. Such a system would support CMC through modeling learners' knowledge and relating it to prerequisite relationships between the single items of knowledge. Chapter 3 documents the implementation of the `C-RATH` prototype system realising the core aspects of the model from Chapter 2. While the proramming is primarily oriented towards easy prototyping and less towards efficientcy and performance, the architecture and the involved design of relational database tables is already oriented towards a final system. Furthermore, this chapter includes the installation and usage of the `C-RATH` system. Since the installation procedure has not changed fundamentally, it is documented here only briefly. Finally, Chapter 4 gives an outlook to further developments of the system. One aim of such further developments is to move forward from the prototype stage to a professional system for "real" application. This includes performance issues as well as interface design or startup assignments. The second central issue with respect to further developments is the theoretical extension to include more general prerequisite structures or models of unobservable skills underlying the observed learners' behaviour.

1.1. The Relational Adaptive Tutoring Hypertext System `RATH`

The Relational Adaptive Tutoring Hypertext System `RATH` was developed by Hockemeyer (1997a; Hockemeyer et al. 1998) connecting the theory of knowledge spaces (Doignon and Falmagne, 1985, 1999; Albert, 1994; Albert and Lukas, 1999) with a relational, Dexter–based hypertext model (Albert and Hockemeyer, 1997).

1.1.1. Overview of Knowledge Space Theory

The theory of knowledge spaces was originally developed by Doignon and Falmagne (1985, 1999) as a foundation for the adaptive, behaviourally oriented assessment of knowledge within a certain domain. Several extensions of this model have been developed, e. g. dealing with non–observable skills underlying the observable answering behaviour (for an overview, see Albert and Lukas, 1999), or with structures between tests (Brandt, Albert, and Hockemeyer, 1999). Since several years, applications of this approach have been extended from the original assessment of knowledge to its training and teaching. Examples are the AdAsTra system (Dowling, Hockemeyer, and Ludwig, 1996), the ALEKS[1] system (Doignon and Falmagne, 1999), and the `RATH`[2] system (Hockemeyer, 1997a; Hockemeyer et al., 1998) which was used as a basis for the software implementation in this project.

If Q is a set of items, the *knowledge state* of a student can be described as the subset of items, this student masters. Due to prerequisite relationships between items, the set of possible knowledge states, the *knowledge space*, is restricted to a subset of the power set of Q. One way to represent such prerequisite relationships is the *surmise relation*. Two items $x, y \in Q$ are in prerequisite relation ($x \sqsubseteq y$) if, from a correct answer to item y, we can surmise a correct answer to item x. Each surmise relation describes a unique knowledge space.

Surmise relations are assumed partial orders on Q. Therefore, they can be illustrated through Hasse diagrams. Figure 1.1 shows such Hasse diagrams of a surmise relation and the corresponding knowledge space. Knowledge spaces corresponding to a surmise relation are closed under union and intersection, i. e. for any two knowledge states S, S', their union ($S \cup S'$) and their intersection ($S \cap S'$) are also knowledge states. Such knowledge spaces are called *quasi–ordinal knowledge spaces.* In a quasi–ordinal knowledge space, there exists, for each item $q \in Q$, a unique minimal knowledge state $C_q \in \mathcal{K}$ containing q. This state C_q is called *clause of q*.

[1]See `http://www.aleks.com` .
[2]See `http://wundt.kfunigraz.ac.at/rath/` .

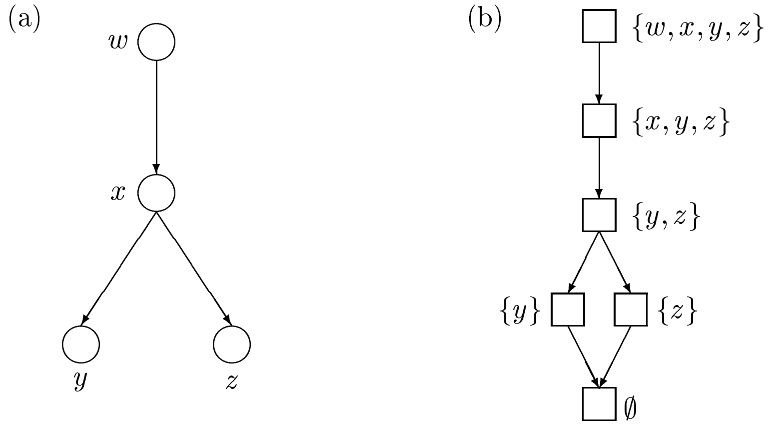

Figure 1.1.: Surmise relation (a) and knowledge states (b) for the problems in Q. (Items are marked by circles and states are marked by squares)

Applying knowledge space theory for tutoring, we obtain the concept of *learning paths* (Falmagne, 1989, 1993) describing students' possible ways from the complete novice (knowledge state $\emptyset$) through the knowledge space to the complete expert (knowledge state Q). We will come back to the concept of learning paths in Section 1.1.3. If, for a knowledge space $\mathcal{K}$, all steps within all learning paths involve a one-item-difference, the knowledge space $\mathcal{K}$ is called *well–graded*.

A more general concept for representing prerequisite relationships has been specified through *surmise systems* (Doignon and Falmagne, 1985). Graphically, they can be illustrated through and–or–graphs. Knowledge spaces corresponding to surmise systems are closed under union but not necessarily under intersection. In this case, there may exist several clauses, i. e. minimal knowledge states C_q, for an item q. This approach has not been applied in RATH, yet. The ALEKS and AdAsTra systems mentioned above apply surmise systems instead of surmise relations.

1.1.2. A Relational, Dexter–based Hypertext Model

Based on hypertext models suggested by Tompa (1989) and by Halasz and Schwartz (1990, 1994), Albert and Hockemeyer (1997) have used a relational notation for formally describing hypertext structures. Within this model, a *hypertext* $H = (C, L)$ consists of a set C of *components* and a set L of *links*. A component $c = (b, S_c, D_c)$ is built by a *base component* b (i. e. the basic unit of information), a set S_c of *source anchors*, and a set D_c of *destination anchors*. The source anchors $s_c \in S_c$ and the destination anchors $d_c \in D_c$ are located on the base component. The sets of all source anchors and destination anchors are denoted by S and D, respectively. A *link* $l \in L$ with $l = ((c, s_c), (c', d_{c'}))$ connects the source anchor s_c located on the component c with the destination anchor $d_{c'}$ located

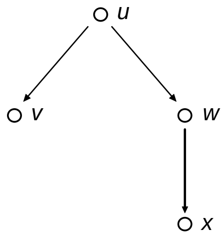

Figure 1.2.: Prerequisite links in a set of components

on the component c'. We also call l a link from c to c'. Based on this formalization of a hypertext, we introduce a binary relation $\vdash \subseteq C \times C$ for a hypertext $H = (C, L)$ by $c \vdash c'$ if and only if there exists a link $l = ((c, s_c), (c', d_{c'})) \in L$.

By distinguishing a subset $P \subset S$ of source anchors we also obtain a distinct subset $L^P \subset L$ by defining $L^P = \{l = ((c, s_c), (c', d_{c'})) \mid s_c \in P\}$. Consequently, we can define a binary *prerequisite link relation* $\vdash^P$ on the set C of components by applying the definition of $\vdash$ to the reduced set L^P of Links. This mechanism can be used to define different *link types*. One example for such link types is the *prerequisite link* in a hypertext tutoring system. Such a prerequisite link from c to c' would specify that, for understanding the content of c, a student must know the content of c'. Figure 1.2 illustrates prerequisite links within a set $C = \{u, v, w, x\}$ of four components.

1.1.3. Connecting knowledge space and hypertext theory

Based on a prerequisite link relation as described above, an adaptive hypertext tutoring system should restrict the student's navigational possibilities to those components they can understand with their existing knowledge. This restriction of navigation can be realized in the framework of knowledge space theory as presented in Section 1.1.1.

If we assume the student's knowledge to comprise the components presented to the student so far, we only have to regard the subsets of components they have visited and/or mastered (cf. Albert and Hockemeyer, 1997).

Because of the restriction of the student's navigation only some subsets of components can be reached in this way. Figure 1.3 shows those subsets of components which may have been presented to a student if we assume the prerequisite link structure from Fig. 1.2.

Taking into account this structure we obtain that there is only a limited number of ways to get from the empty set (or another starting point) to the set of all components (or another learning goal. For a Hypertext $H = (C, L)$ and a prerequisite link relation

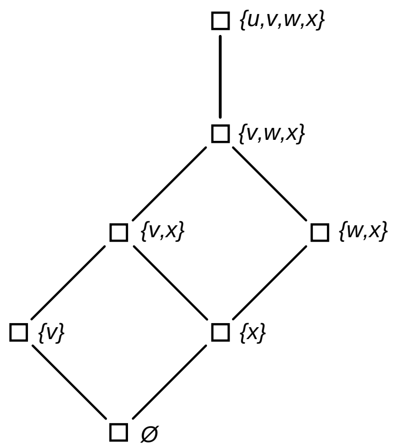

Figure 1.3.: Structure according to the prerequisite relation above

$\vdash^P$, a sequence $X_0, X_1, \ldots, X_n$ of subsets of components is called a *learning path* if and only if the following two conditions hold: (i) For any $i = 0, 1, \ldots, n$, for any $c \in X_i$, and for any $c' \in C$, if $c \vdash^P c'$ then $c' \in X_i$. (ii) For any $i = 1, 2, \ldots, n$ there exists a $c \in C \setminus X_{i-1}$ such that $X_u = X_{i-1} \cup \{c\}$. Figure 1.4 shows one possible learning path from the empty set to the complete set in the structure from Figure 1.3. In this example there exist three possible paths. This notion of a learning path also corresponds to that from knowledge space theory mentioned in Section 1.1.1 above.

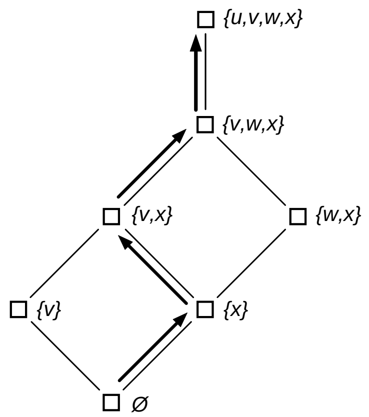

Figure 1.4.: One possible path through the structure from Figure 1.3

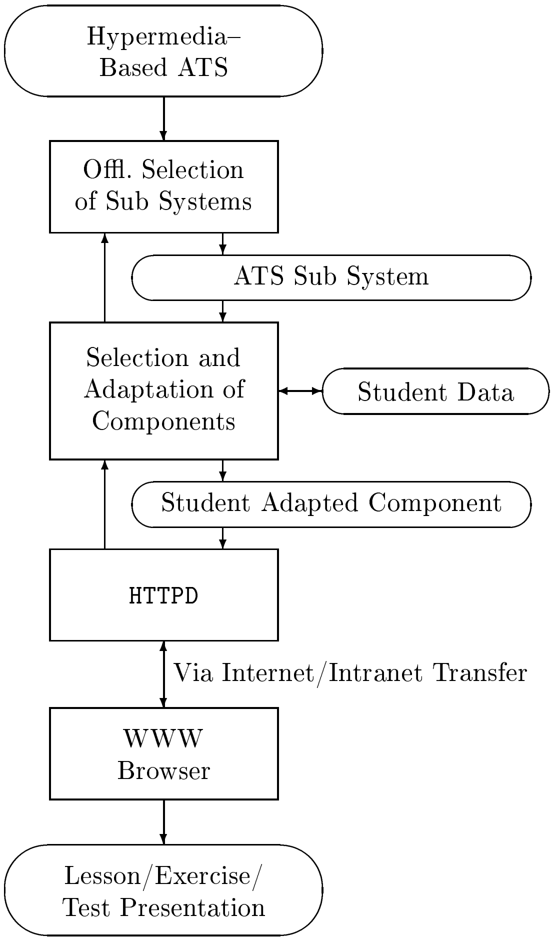

Figure 1.5.: RATH architecture

1.1.4. Realisation of the RATH System

The relational notation of both, knowledge space theory and the hypertext model strongly supports the application of relational database theory and a corresponding — professional and very efficient — relational database management system. As described by Albert, Hockemeyer, and Held (1997), the formal concepts from both models described in the previos sections can be implemented as relational database tables in a straightforward manner. This resulted in a system architecture for RATH as depicted in Figure 1.5. The entire course builds a hypermedia–based adaptive tutoring system (ATS). From this, an ATS sub system may be pre–selected, e. g. in case the learners are required to acquire only a part of the course contents. Both, the link structure of this ATS sub system as well as data on the learners' progress are stored in a relational database system. In the learning process, the learner requests new documents through the browser–HTTPD connection. The HTTPD (web server software) responds to such requests by selecting and adapting (i. e. filtering) documents with respect to the stored information on the learner's knowledge.

1.2. Computer–Mediated Communication

Numerous systems for computer–mediated communication (CMC) have been developed which, in general, restrict themselves to providing technical support for synchronous and/or asynchronous distant communication. Two approaches going beyond this pure technical aspects are summarised below.

1.2.1. Multiple Student Modeling

Hoppe (1995) developed a formalisation for describing students' knowledge by means of logical predicates. From a matematical viewpoint, these predicates can be regarded and written as relations between the domain sets of the predicates' parameters. For example, the predicate $\mathsf{knows}(s,t)$ describing that a learner s knows a topic t can also be formalised through a relation $K \subseteq S \times T$ where S and T are the sets of all learners s and topics t, respectively, by simply defining the relation K through

$$(s,t) \in K \iff \mathsf{knows}(s,t).$$

By this reformalisation, the concepts of Hoppe can easily be connected to the relationally formalised theories of knowledge spaces and of hypertext structures.

Based on this formalisation of a single learner's knowledge, Hoppe introduces formal descriptions for several aspects of collaborative learning. With respect to group problem solving, e. g., he describes a problem selection criterion which appraises a problem p as adequate for a learning group $\{s_1, s_2, ldots, s_n\}$ if none of the group members $s_i, i = 1, \ldots, n$ is capable of solving the problem p alone but the the aggregated knowledge of the whole group is sufficient for finding a solution.

A second, important aspect of collaborative learning which is also in our project's focus is the mutual support between peers within a course. Hoppe suggests a rule that a learner s_1 can help a learner s_2 on a topic t if $(s_1, t) \in K$ and $(s_2, t) \notin K$.

1.2.2. The Intelligent Distributed Collaborative Learning Environment iDCLE

The second CMC approach looking promising for us was the *Intelligent Distributed Collaborative Learning Environment* (iDCLE) developed by Inaba and Okamoto (1995, 1997). They have developed dialogue models including rules for assessing the state of a dialogue or discussion. In their system, learners have to specify the function (or intention) of their utterance (e. g. proposition, question, answer, agreement, disagreement, explanation, or supplement) which then is used for assessing the discussion process. Inaba and Okamoto have described four different modes of communication. For each of these modes, they have identified the different occuring dialogue states and the possibile state transitions depending on the utterance intentions specified.

For our purpose, the most important concept currently is the intention of an utterance describing the relation between the utterance and the referred document which may be a course document or simply another utterance.

2. Modelling a Collaborative Adaptive Tutoring Hypertext System

In developing a formal model as a basis for the `C-RATH` system, we have focused on two aspects: (i) integrating the concept of utterances into our relational hypertext model and (ii) selecting adequate peers to be invited to join a dialogue.

2.1. Integrating Utterances into the Relational Hypertext and the Prerequisite Structure

Integrating utterances into a relational hypertext course can be investigated with two different perspectives: the formal perspective describes the integration of the utterance concept into the relational hypertext model while the educational perspective regards the question under which conditions learners' utterances could or should be made available to their peers.

2.1.1. Formal Perspective of the Integration

Utterances made by learners with respect to course documents or to other utterances can easily be integrated into the relational hypertext model described in Section 1.1.2 by regarding utterances themselves as documents.

Let us assume that our set D of destination anchors contains an anchor $\perp$ such that $\perp \in D_c$ for all components $c \in C$. Similarly let us assume a source anchor $\top$ such that $\top \in A_c$ for all $c \in C$. A new utterance u made with respect to a component $c \in C$ can be integrated into a hypertext $H = (C, L)$ by inserting the utterance u into the set C of components and inserting the reference link $l_u = ((u, \top), (c, \perp))$ into the link set L. Furthermore, let us assume a predefined set I of utterance intentions. We can now define a subset $L^i \subseteq L$ for each predifined intention $i \in I$ such that if an utterance u is made by its author with the specified intention $i \in I$, the corresponding reference link l_u is also inserted into the subset L^i of links.

Another topic in integrating new utterances into the hypertext link structure — especially with respect to the `RATH` system — is the positioning into the prerequisite structure of components. In a first, automatic step this could be done extending the set L^P of prerequisite links by the set

$$L_u^P = \left\{ l_u = ((u, \top), (c, d_c)) \mid \exists c' \in C : l_{c'} = ((c', s_{c'}), (c, d_c)) \in L^P \right\}. \tag{2.1}$$

2.1.2. Educational Perspective of the Integration

In a first step, it is important to distinguish utterances from the original course materials. This distinction must necessarily take place on an administrative level, i. e. the *base component* must also contain information, e. g., on whether it is an utterance or a course document, and who is the author of the utterance. One may, however, also discus a distinction in representation, e. g. by displaying the author's name when presenting an utterance to the reader.

In a second step, editing policies have to be decided upon, e. g. whether the tutor or the author may (re–) edit an utterance, or whether utterances are moderated a priori or may be moderated a posteriori by the tutor.

Since the project reported here is primarily concerned with the formal aspects of adaptive tutoring systems, the educational perspective has not been persecuted in more detail. This will be part of future research and development.

2.2. Selecting Appropriate Peers

Hoppe (1995) has already refined the selection of peers for a dialogue invitation by his `can_help` predicate which considers all peers as appropriate who know the topic taught in the document referred to by the utterance. In case of distant learning systems, however, this set of peers may grow very large. An even more important point to consider is the fact that advanced learners within a course would get a high amount of questions etc. of which many may seem to them being on a very basic level. This may well result in boredom for these advance learners and, as a consequence, in a reluctance to answer to questions at all. This issue also resembles a well–known experience from the USENET where advanced readers of a newsgroup cease answering questions they may have seen answered already dozens of time.

Our solution to this problem is a selection of appropriate peers for forwarding them questions (and other utterances) not only based on the question whether or not they have the knowledge necessary but also based on the question how far they have already advanced beyond the level of the document referred to.

Let $\mathcal{K}$ be the knowledge space on some course, and let $\mathcal{L} \subseteq \mathcal{K}$ be the set of all knowledge states currently occurring among the groups of learners within this course. Furthermore, we define a mapping $\delta : \mathcal{K} \times Q \longrightarrow \mathbb{N}$ such that, for an item $q \in Q$ and a knowledge state $K \in \mathcal{K}$, we obtain

$$\delta(K, q) = \begin{cases} \infty & \text{for} \quad q \notin K \\ \max\{|K \Delta K'| \text{ for } q \in K' \text{ and } K' \subseteq K\} & \text{for} \quad q \in K \end{cases}$$

where Δ is the symmetric set difference (Doignon and Falmagne, 1999). Based on this, we can define a mapping $\mathcal{P} : Q \longrightarrow \wp(\mathcal{L})$ which assigns to each item q in the domain of knowledge the set $\mathcal{P}(q)$ of most appropriate peer knowledge states for answering questions with respect to this item q if "most appropriate" is defined as "having acquired the knowledge q probably most recently". This is achieved by defining $\mathcal{P}$ as

$$\mathcal{P}(q) = \{K \in \mathcal{L} \mid \delta(K, q) \text{ is minimal}\}. \tag{2.2}$$

Remark: The above definitions of the function δ and the set $\mathcal{P}$ of most appropriate peer knowledge states are closely related to the *neighbourhood* and *fringe* concepts introduced by Falmagne and Doignon (1988, see also Doignon 1994; Doignon and Falmagne 1999) and extended by Hockemeyer (1997b). In particular, we obtain

$$\delta(K, q) = \min\{\epsilon \mid q \in F^i(K, \epsilon)\} - 1 \tag{2.3}$$

for all $K \in \mathcal{K}$ and $q \in K$, i. e. $\epsilon = \delta(K, q) + 1$ denotes the minimal value for ϵ such that q is an element of the inner ϵ–fringe of K. In terms of neighbourhood, we obtain

$$\mathcal{P}(q) \subseteq N(\sigma(q), \delta(K, q)) \tag{2.4}$$

where $\sigma(q)$ denotes the set of all clauses C_q of q, i. e. all peers covered by some knowledge state $K \in \mathcal{P}(q)$ are in a knowledge state which is a $(q, \delta(K, q))$–neighbour of the knowledge state of someone, who has just learned item q. In case of well–graded knowledge spaces, this is equivalent to to $K \in \mathcal{P}(q)$ denoting knowledge states which, for a minimal ϵ, are (q, ϵ)–neighbours of the knowledge state of some learner who is about to learn q.

3. Implementation

3.1. Architecture

The architecture of the new `C-RATH` system is strongly influenced by the well approved `RATH` architecture. Figure 3.1 depicts the `RATH` architecture in a way which is in comparison to Fig. 1.5 reduced to the components used "online": The client browser requests through the WWW port documents from the HTTPD. Through its CGI interface, the HTTPD retrieves the requested documents and filters them based on information obtained from the Oracle relational database. Finally, the filtered documents are served through the internet to the requesting client to be presented in the browser window.

Moving on to the new `C-RATH` system whose architecture is shown in Fig. 3.2, the most important change with respect to the consequences towards the system's architecture is the need to inform learners synchronously about certain incoming utterances. This includes especially the notification that a learner's utterance has been answered or the information that a learner has been considered being most appropriate for reacting to a peer's utterance. As a consequence, we need a possibility for the server to start an

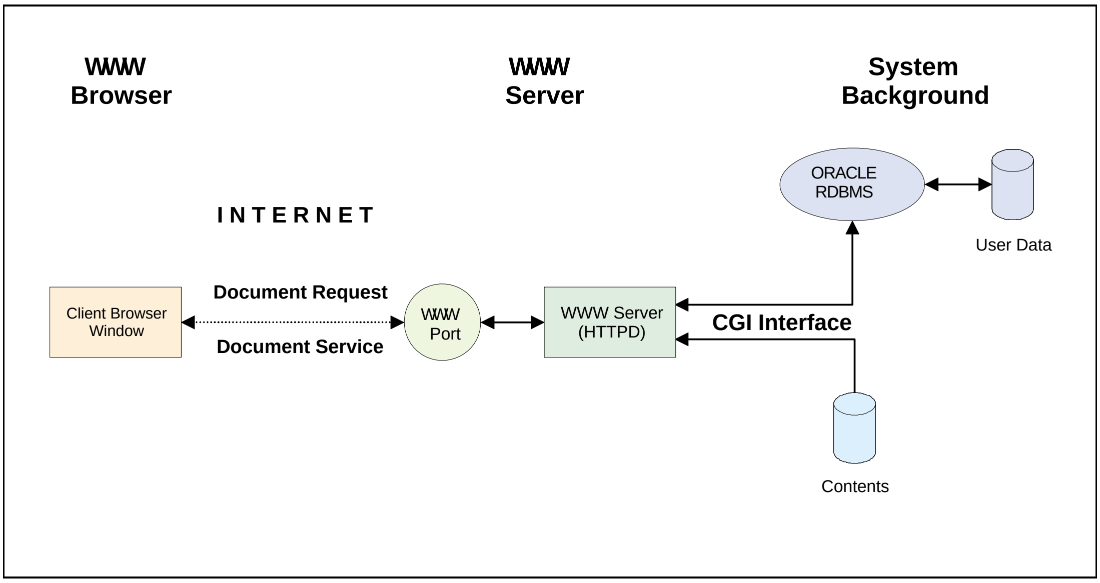

Figure 3.1.: Simplified structure of the `RATH` system

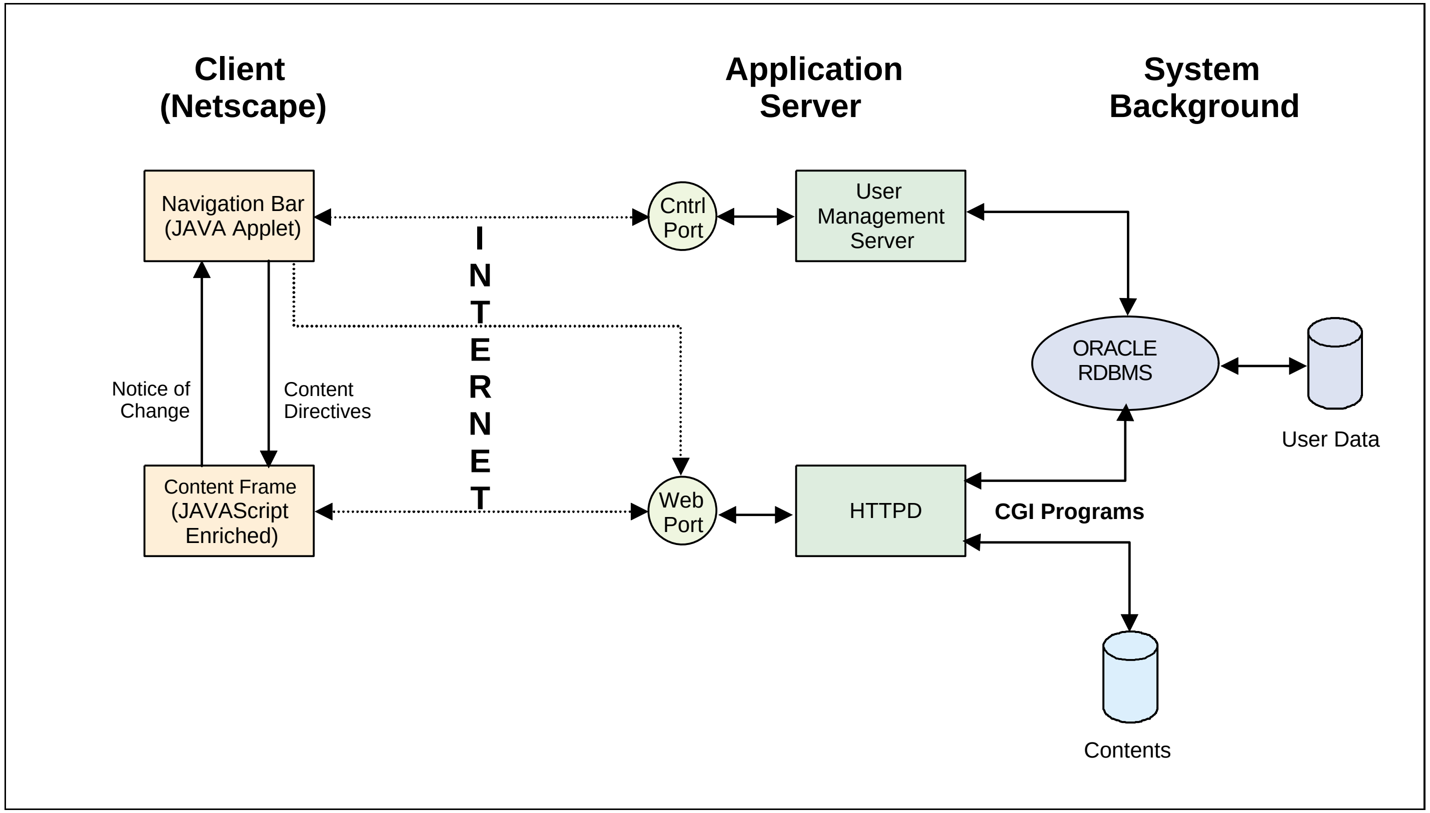

Figure 3.2.: `C-RATH` architecture

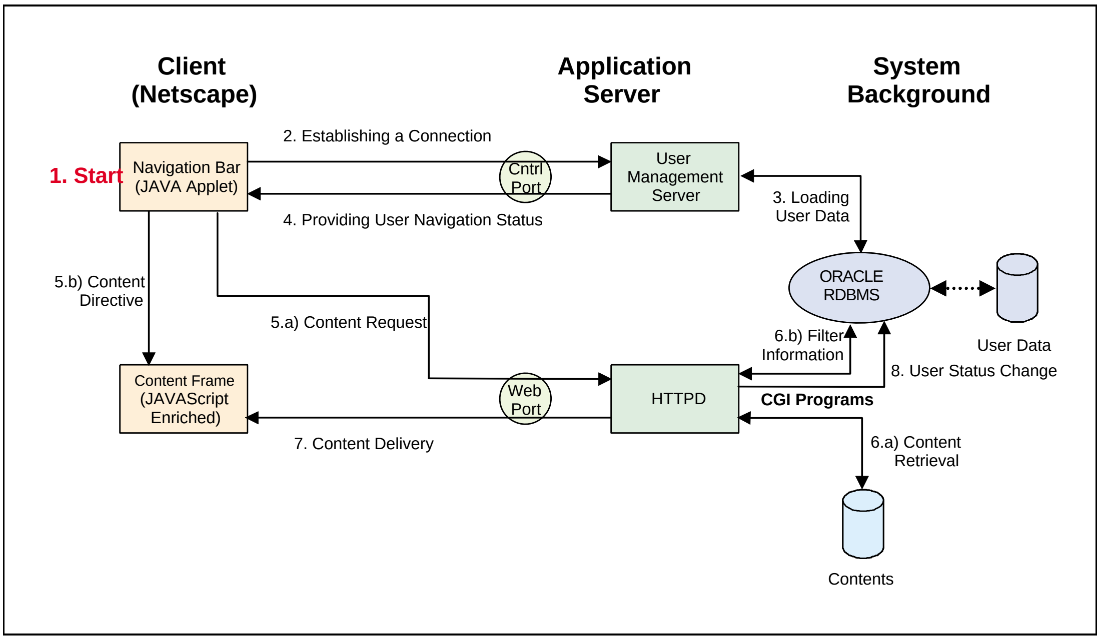

Figure 3.3.: Starting the C-RATH application

interaction with the client - a possibility which is only very poorly supported by the HTTP protocol. Planning the C-RATH system, this led to introducing a second communication track.

On the client side, we now have a dedicated navigation frame within the browser window where the *navigation bar* is realised through a JAVA applet which connects with a *user management server* through a predefined control port. In case of user actions in one of the client frames either the navigation applet issues a content directive for the content frame, or the content frame informs the navigation applet through included JavaScript routines about the learner's link selections.

The figures 3.3–3.7 show certain aspects of the client server communication in more detail.

When the learner starts the C-RATH course (Fig. 3.3), the navigation applet is called first. It establishes a connection to the user management server which in turn retrieves the corresponding user data from the database system. It provides the navigation applet with information on the learner's navigation status stored from the last interaction. Based on this, the navigation applet requests some content document from the HTTPD and directs it to the content frame. The HTTPD retrieves filter information and contents through CGI programs and delivers the filtered content to the client's content frame. Finally, the HTTPD stores information on the user's status change in the database system.

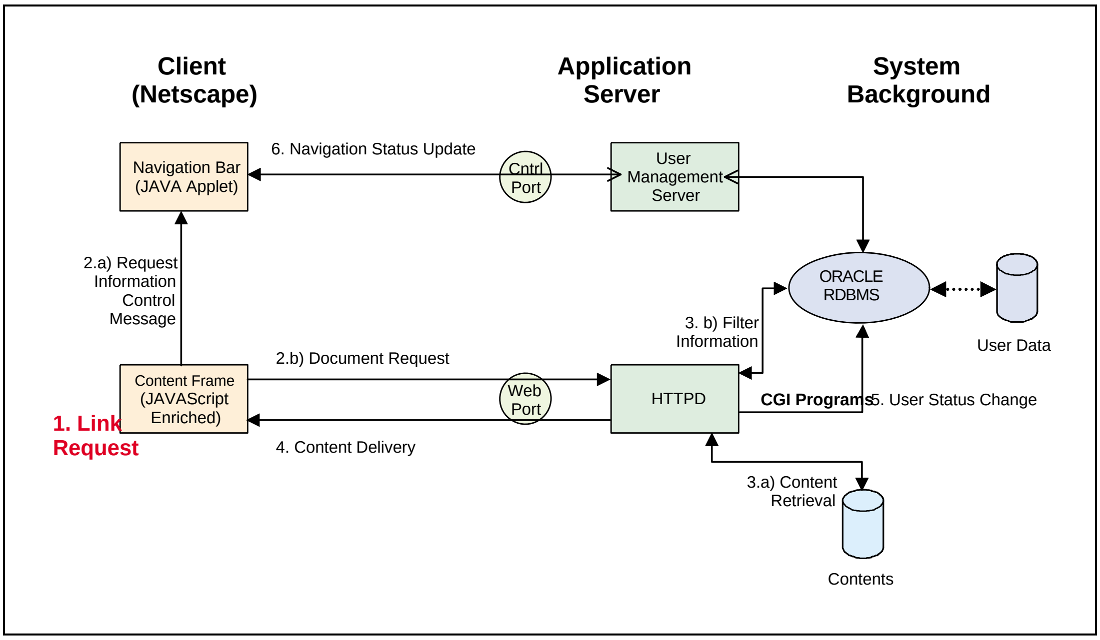

Figure 3.4.: Selecting a link in the content frame

If the learner selects a link in the content frame (see Fig. 3.4), its JavaScript routines send a control message informing the navigation applet about this request. Furthermore, the request itself is sent to the HTTPD which treats it the way already described above. Finally, the navigation applet updates the navigation status stored in the database system through its connection with the user management server.

On the learner's selection of the "new utterance" button (see Fig. 3.5), the navigation bar launches an *utterance editor* frame after backing up the user's current navigation status. When the user finishes the editing process, the utterance editor frame is closed, and the utterance is stored through the HTTPD in the set of contents and registered in the database. Afterwards, the user's navigation status is restored as already described in the start case.

When a learner selects a document for which utterances are available or when s/he is deemed as a most appropriate peer for reacting on some other learner's utterance, this is indicated in the navigation frame based on its regular status update (see below). If a user selects an utterance which is not linked to the currently read document but suggested to him/her as most appropriate peer this is considered as a type of interrupt (Fig. 3.6). Therefore, the navigation applet again backups the current navigation status before requesting the respective utterance document as described in the start case. While the learner is dealing with this utterance interrupt, the navigation applet provides an

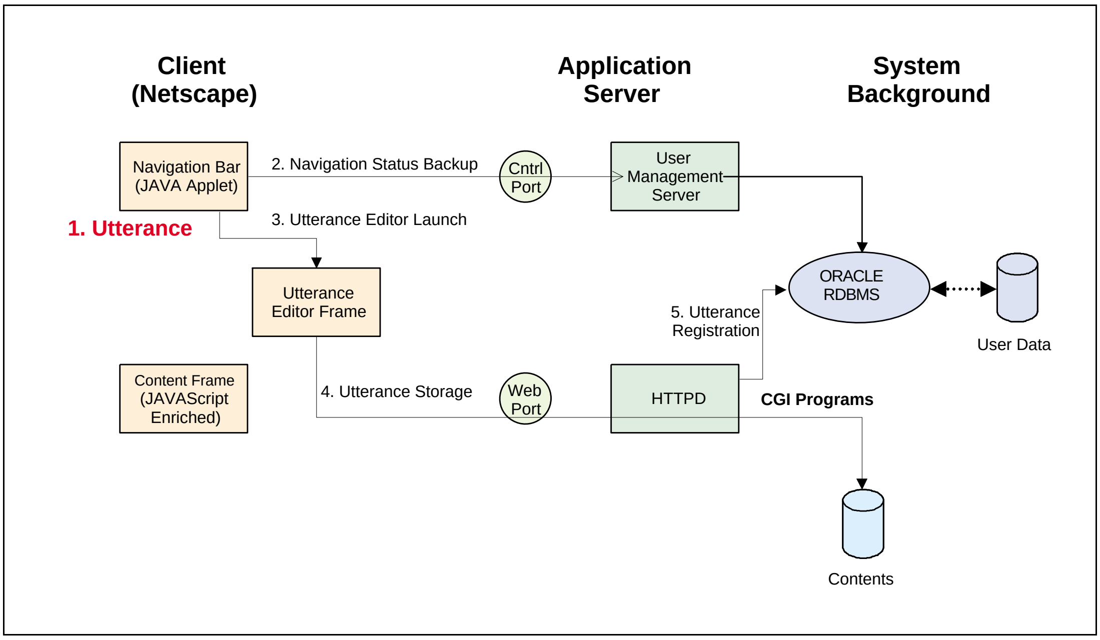

Figure 3.5.: Creating a new utterance

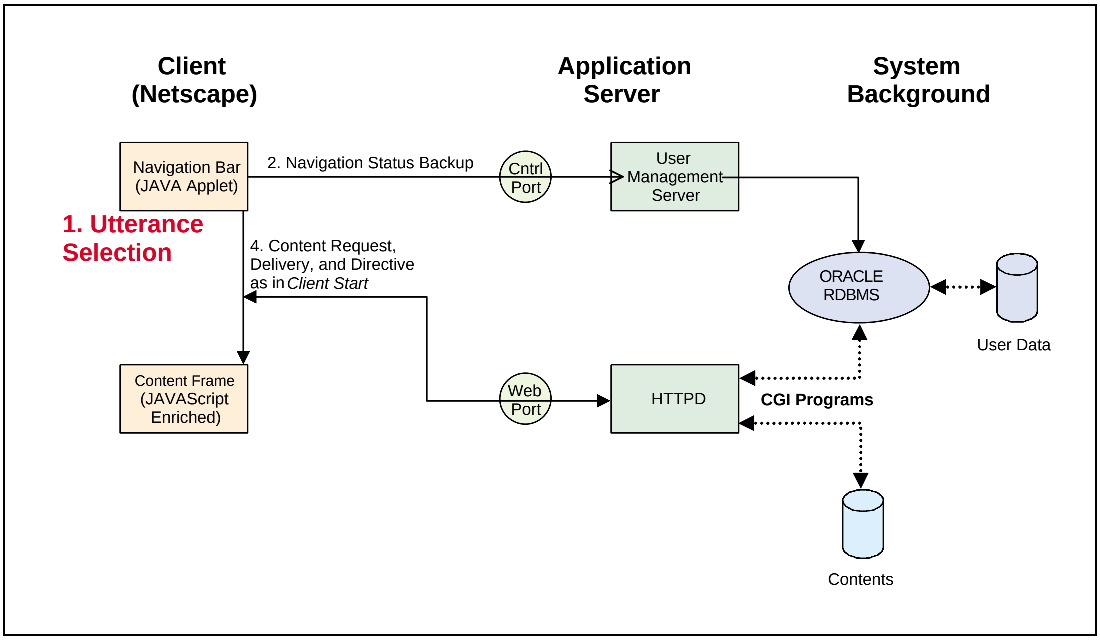

Figure 3.6.: Selecting an utterance

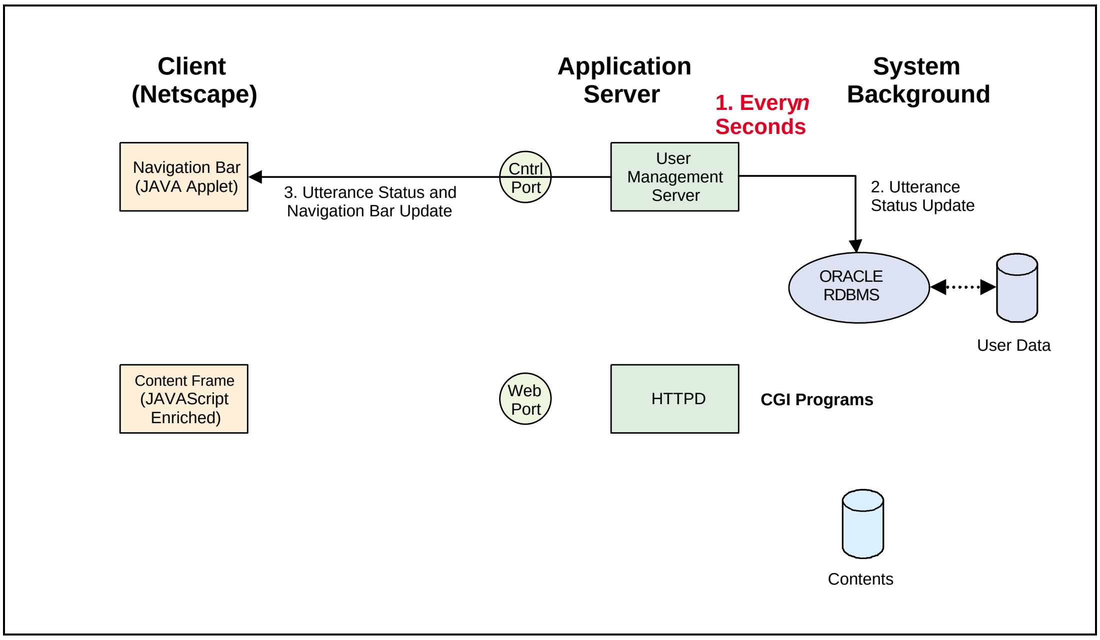

Figure 3.7.: Regular status update

additional return button to allow the learner to return to his/her regular learning process at any time utilising the navigation status backup made before.

Finally, the user management server starts on a regular basis a navigation status update (Fig. 3.7). In this process, it requests the current status of awaiting utterances and informs the navigation applet to initiate the navigation bar's extension by utterance buttons if appropriate.

3.2. Database Relations

In this section, the major database tables used in `RATH` (Hockemeyer, 1997a, Appendix B) will be briefly repeated before describing the changes for the new `C-RATH` system. In many of these tables columns Xid are used for unique numerical identifiers for X. For any table containing a certain column Xid, this column denotes the same type of objects and vice versa. For example, the column sid (see below) denotes the source anchor identifier in all tables where it occurs and, vice versa, whenever a table column denotes a source anchor, it is called sid.

3.2.1. Database Relations Used in RATH

On the document side, we start with a table doc(did, filename, title). Anchors on these documents are specified through the tables source(sid, did) and dest(aid, did, name), and described in more detail through the tables source_prop(sid, dir, sprop) and dest_prop(aid, dir, dprop), respectively, where dir specifies the direction of the described property (this part is influenced by the HTML language). Finally, the table link(lid, sid, aid) contains the links themselves.

The information on the learners, on the other side, are contained in two tables pupil(puid, account, name) and performance(puid, did, when). Since RATH does not include an assessment of the learner, a pupil's performance is defined by the set of documents already visited. Note that, currently, an exercise document is regarded as visited after it has been solved correctly. These exercise documents are identified through an extra table forms(did), i. e. actually through a list of the respective document identifiers.

The values for each of the Xid identification numbers are selected through Oracle's *sequences* in order to ensure uniqueness.

3.2.2. New Database Relations Introduced in C-RATH

A first extension of the database is the more detailed description of the course documents. A first part of it is the extension of the table doc(did, filename, title, dtype, author, when). Currently used dtype values are text, example, exercise, and utterance. By these dtype entries, also the forms table from the RATH system has become obsolete.

The major extension of the database design then is a new table utterance(did, rdid, utype) whose entries denote that document did is an utterance of type utype about the document rdid (the name of the rdid column is an exception to the general principle of naming all document identifier columns did induced by the necessity of unique column names within each table.

Finally, a new table user_settings(puid, variable, value) has been introduced. Currently, this table is only used for the navigation status backups (see Section 3.1). However, it has been designed to also include possible future user preference settings (see Chapter 4.3 below).

3.3. Installation and Usage

The installation of the C-RATH system is basically the same as in the RATH system — only the word *rath* has to be exchanged by C-RATH (in the proper case) at several places. Since the installation has been described in depth in the RATH manual (Hockemeyer, 1997a), it can be omitted here.

With respect to the usage, a number of things have changed for the course tutor as well as for the learner. On the side of the course tutor (or administrator), a major change is the enrolement of new learners. In the original RATH system, each new learner had to be enroled by some administrator. Later on, a facility for self–enroling was introduced. Based on our experiences with this facility, the enrolement by adminsitrator has now been omitted completely. Since the learner can also retrieve their learning progress themselves, there is currently no real need for the administrator user type; however, it will be kept for future changes which are likely to need it again (see below, Chapter 4.2).

From the learner's side, the user interface has changed quite a lot — however primarily on the optical side. The navigation bars have been retracted from the single documents and have been put into the new, dedicated navigation frame (the navigation applet described in Section 3.1).

4. Further Directions

In this chapter, a number of further directions in research and development of the `C-RATH` system are named.

4.1. A Dedicated Server for Educational Content Provision

A primarily prformance–oriented improvement is an architectural change of the content serving component. Figure 4.1 shows a slightly changed system structure where the HTTPD and its CGI interface is replaced by a dedicated educational *content server*.
This content server has several advantages in comparison to the current solution. Currently, each request launches a filter program through the CGI interface which in turn builds up a connection to the database server. Since both sub processes are very time consuming, usage of a dedicated Server with a steady connection to the database system is expected to strongly accelerate the document serving. Furthermore, such a dedicated content server could also build a steady connection to the user management server which

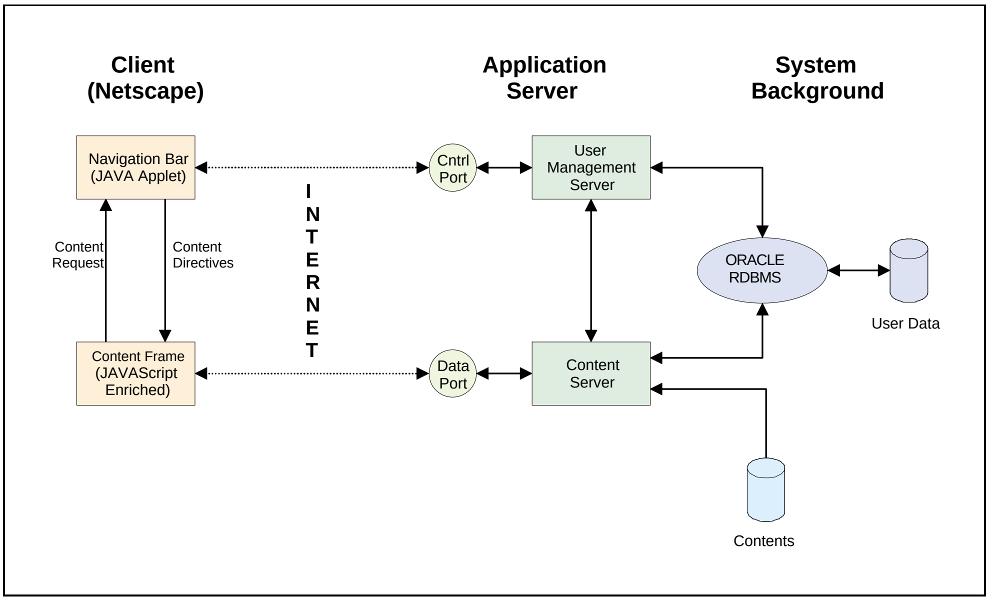

Figure 4.1.: Prerequisite links in a set of components

would simplify and accelerate the user navigation and utterance status updates.

4.2. Moderating Utterances

As already mentioned in Section 2.1.2 above, a dialogue system like this introduced with `C-RATH` should provide — from an educational perspective — a mechanism for moderation, i. e. approval, correction, or rejection of learners' utterance in order to prevent the system's users from being mislead by peers' utterances.

4.3. Developing a *Production System*

Finally, the `C-RATH` system shall be developed from its current prototype status to a "real use" system which would also include commercial marketing in order to gain some return of investment. Such progress would include, e. g. the support of different, independent courses in one system, the — currently omitted — control of enrolment, the introduction of knowledge assessments as criterion for learner's performance instead of the downloaded documents, and enhancements of the unser interface like the progress report.

Bibliography

Albert, D. (Ed.). (1994). *Knowledge structures.* New York: Springer Verlag.

Albert, D. & Hockemeyer, C. (1997). Adaptive and dynamic hypertext tutoring systems based on knowledge space theory. In B. du Boulay & R. Mizoguchi (Eds.), *Artificial Intelligence in Education: Knowledge and Media in Learning Systems* (pp. 553–555). Amsterdam: IOS Press.

Albert, D., Hockemeyer, C., & Held, T. (1997). *A relational model for hypertext structures.* Unpublished Manuscript.

Albert, D. & Lukas, J. (Eds.). (1999). *Knowledge spaces: Theories, empirical research, applications.* Mahwah, NJ: Lawrence Erlbaum Associates.

Brandt, S., Albert, D., & Hockemeyer, C. (1999). Surmise relations between tests - preliminary results of the mathematical modelling. *Electronic Notes in Discrete Mathematics*, *2*.

Dillenbourg, P. (1999). Introduction: What do you mean by "collaborative learning". In P. Dillenbourg (Ed.), *Collaborative Learning: Cognitive and Computational Approaches* (pp. 1–19). Amsterdam: Pergamon.

Doignon, J.-P. (1994). Probabilistic assessment of knowledge. In D. Albert (Ed.), *Knowledge Structures* (pp. 1–56). New York: Springer Verlag.

Doignon, J.-P. & Falmagne, J.-C. (1985). Spaces for the assessment of knowledge. *International Journal of Man-Machine Studies*, *23*, 175–196.

Doignon, J.-P. & Falmagne, J.-C. (1999). *Knowledge spaces.* Berlin: Springer–Verlag.

Dowling, C. E., Hockemeyer, C., & Ludwig, A. H. (1996). Adaptive assessment and training using the neighbourhood of knowledge states. In C. Frasson, G. Gauthier, & A. Lesgold (Eds.), *Intelligent Tutoring Systems* (pp. 578–586). Berlin: Springer Verlag.

Falmagne, J.-C. (1989). A latent trait theory via stochastic learning theory for a knowledge space. *Psychometrika*, *53*, 283–303.

Falmagne, J.-C. (1993). Stochastic learning paths in a knowledge structure. *Journal of Mathematical Psychology*, *37*, 489–512.

Falmagne, J.-C. & Doignon, J.-P. (1988). A Markovian procedure for assessing the state of a system. *Journal of Mathematical Psychology*, *32*, 232–258.

Halasz, F. & Schwartz, M. (1990). The Dexter hypertext reference model. In J. Moline, D. Benigni, & J. Baronas (Eds.), *Proceedings of the Hypertext Standardization Workshop* (pp. 95–133). Gaithersburg, MD 20899: National Institute of Standards and Technology.

Halasz, F. & Schwartz, M. (1994). The Dexter hypertext reference model. *Communications of the ACM*, *37*(2), 30–39.

Held, T. (1993). *Establishment and empirical validation of problem structures based on domain specific skills and textual properties.* Dissertation, Universität Heidelberg, Germany.

Hockemeyer, C. (1997a). *RATH — a relational adaptive tutoring hypertext WWW–environment.* (Tech. Rep. 1997/3), Institut für Psychologie, Karl–Franzens–Universität Graz, Austria. To be obtained from Cord.Hockemeyer@kfunigraz.ac.at.

Hockemeyer, C. (1997b). Using the basis of a knowledge space for determining the fringe of a knowledge state. *Journal of Mathematical Psychology*, *41*, 275–279.

Hockemeyer, C., Held, T., & Albert, D. (1998). RATH — a relational adaptive tutoring hypertext WWW–environment based on knowledge space theory. In C. Alvegård (Ed.), *CALISCE'98: Proceedings of the Fourth International Conference on Computer Aided Learning in Science and Engineering* (pp. 417–423). Göteborg, Sweden: Chalmers University of Technology.

Hoppe, H. U. (1995). The use of multiple student modeling to parameterize group learning. In J. Greer (Ed.), *Artificial Intelligence in Education, 1995* (pp. 234–241). Charlottesville, VA: Association for the Advancement of Computing in Education (AACE).

Inaba, A. & Okamoto, T. (1995). The network discussion supporting system embedded computer coordinator at the distributed places. *Educational Technology Research*, *18*, 17–24.

Inaba, A. & Okamoto, T. (1997). The intelligent discussion coordinating system for effective collaborative learning. In T. Okamoto & P. Dillenbourg (Eds.), *Collaborative Learning/Working Support System with Networking* (pp. 26–33). Kobe, Japan. Workshop at the 8th World Conference on Artificial Intelligence in Education AI–ED 97.

Ottmann, T. & Tomek, I. (Eds.). (1998). *ED–Media/ED–Telecom'98*, AACE. Charlottesville, VA: Association for the Advanvement of Computers in Education.

Tompa, F. W. M. (1989). A data model for flexible hypertext database systems. *ACM Transactions on Information Systems*, *7*(1), 85–100.

Veerman, A. L., Andriessen, J. E. B., & Kanselaar, G. (1999). Collaborative learning through computer–mediated argumentation. In *Proceedings of the CSCL'99* (pp. 640–650). Stanford University.

Appendix

A. Example Course Information

For the demonstration of applicability of the system conceptualised in Chapter 2 and implemented according to Chapter 3, we have reused the tiny course on elementary probability theory already applied in the original `RATH` system which is base don the work of Held (1993). Besides a few corrections, four major changes have been made:

1. As already mentioned, the navigation buttons have been moved from the single documents to a special navigation frame. This changes the look of the concrete documents.

2. In order to support different types of learners (e. g. theory–oriented vs. example–oriented), the examples do not any longer have their corresponding lessons as prerequisites but they share the prerequisite with these corresponging lessons. This is supported by theoretical definition and examples being parallel subsections of the respective topic section.

3. The exercises which have the purpose of test problems evaluating the learner's progress have been moved from the separate section at the end of the course to be spread between the lessons such that lessons *and* exercises could be done in the linear order as they are specified in Appendix B.

4. Finally, the prerequisite structure within the course has been slightly corrected where in the `RATH` system additional prerequisites had been introduced falsely in the process of combining the structures of teaching contents and of training problems. Figure A.1 depicts this corrected structure.

The production of the course itself was performed the same way as it had been done in the `RATH` system: The lessons and exercises were written using LaTeX; afterwards, this text was translated into HTML using the LaTeX2HTML[1] program, and the meta–information needed for providing the adaptivity (i. e. the prerequisite links) were added manually.

[1] See `http://www-dsed.llnl.gov/files/programs/unix/latex2html`.

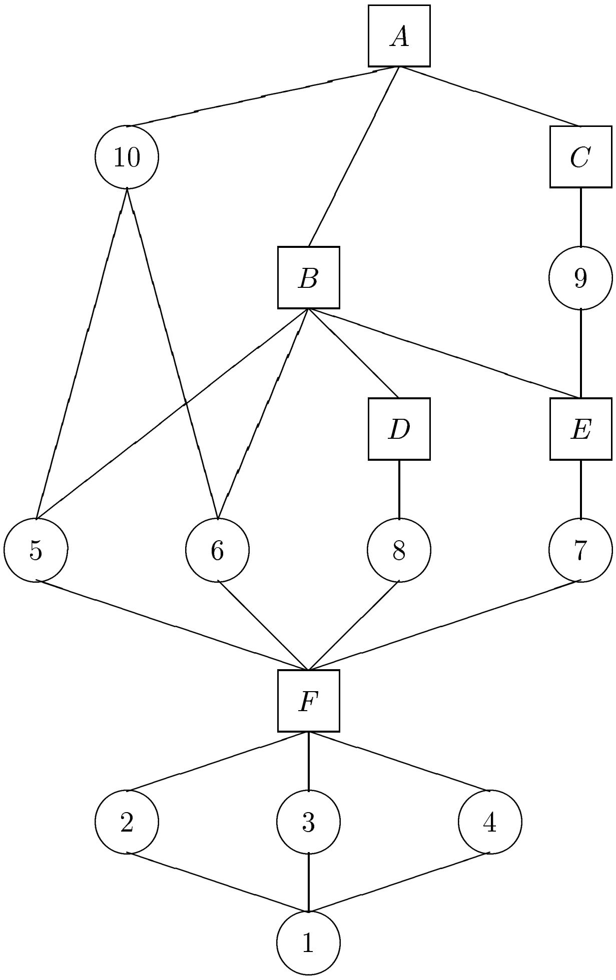

Figure A.1.: Revised structure of teaching contents and training problems

B. Example Course Contents

The following text contains the teaching material of the `RATH` prototype. This material was composed together with Gerhard Hermann using the results of prior research by Held (1993). In the hypertext version, each section constitutes an own document.

B.1. Elementary Events

Theoretical explanation

Experiments which may have different outcomes under the same conditions are called *random experiments.* Throwing a dice, for example, may result in different numbers between one and six.

A random experiment may have a finite number of possible outcomes ω_1,ω_2,..., ω_n ($n \in \mathbb{N}$). The set $\Omega = \{\omega_1,\omega_2, \ldots ,\omega_n\}$ is called the *space of outcomes.*

Example 1: Drawing balls from an urn

Condition: An urn contains two balls, one is white and the other is green.
Random experiment: Drawing one ball.
Determine the number of outcomes.

Solution:
Possible Results:
ω_1 = "white ball"
ω_2 = "green ball"

Space of outcomes:
$\Omega = \{\omega_1, \omega_2\} \Longrightarrow$ **The number of outcomes is $|\boldsymbol{\Omega}| = 2$.**

Example 2: Throwing a dice

Condition: A dice usually has six sides which have one to six points.
Random experiment: Throwing the dice and reading the number of points on its upper side.
Determine the number of outcomes.

Solution:
ω_1 = "There is one point on the dice's upper side."

ω_2 = "There are two points on the dice's upper side."
ω_3 = "There are three points on the dice's upper side."
ω_4 = "There are four points on the dice's upper side."
ω_5 = "There are five points on the dice's upper side."
ω_6 = "There are six points on the dice's upper side."

Space of outcomes:
$\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6\}$
$\Longrightarrow$ **The number of outcomes is $|\boldsymbol{\Omega}| = \mathbf{6}$.**

Example 3: Taking a floppy disk from a box

Condition: A box contains three floppies. One of them is defective.
Random experiment: Taking one floppy from the box.
Determine the number of outcomes.

Solution:
Possible Results:
ω_1 = "Intact floppy"
ω_2 = "Defective floppy"

Space of outcomes:
$\Omega = \{\omega_1, \omega_2\} \Longrightarrow$ **The number of outcomes is $|\boldsymbol{\Omega}| = \mathbf{2}$.**

B.2. Events

Let Ω be a space of outcomes. Any subset $A \subseteq \Omega$ is called *event*. The set of all events is the set of all subsets (or the *power set* $\wp(\Omega)$) of Ω. It is called *event space*. Note that the empty set $\emptyset$ and the space of outcomes Ω are also subsets of the event space Ω and, therefore, events.

An event A has happened if the outcome of the experiment is an element of A and, vice versa, A has not happened if the outcome is not an element of A.

Let $\Omega = \{\omega_1, \omega_2, \dots, \omega_n\}$, $n \in \mathbb{N}$, be a space of outcomes. The events $\{\omega_i\}$, $i = 1, 2, \dots, n$, are called *elementary events* of the event space $\wp(\Omega)$. For a given event $A \in \wp(\Omega)$, the elementary events $\{\omega_i\}$ that fulfil the condition $\omega_i \in A$ are called *convenient for A*.

B.3. Determining the number of convenient events

Theoretical explanation

Let Ω be a set of possible outcomes for a random experiment, and let A be an event described by "the outcome has the property Φ". Then the event A contains all those outcomes $\omega \in \Omega$ that fulfil the specified condition, i. e. that have the property Φ.

Example 1: Drawing balls from an urn

An urn contains two green balls and two white balls.
A green ball is drawn.
Determine the number of convenient elementary events.

Solution:
Event A: "A green ball is drawn."

Convenient elemtary events for the event A are all green balls:
$A = \{green, green\} \implies |A| = 1$
The number of convenient elementary events is 1.

Example 2: Throwing a dice

A dice is thrown.
On its upper side, it shows an even number of points.
Determine the number of convenient elementary events.

Solution:
Event A: "The number of points on the upper side is even."

Convenient elementary events for the event A are the even numbers between one and six:
$A = \{2, 4, 6\} \implies |A| = 3$
The number of convenient elementary events is 3.

Example 3: Taking a floppy disk out of a box

In a box there are two floppies. One is intact, the other is defective.
An intact floppy is taken out of the box.
Determine the number of convenient elemtary outcomes

Solution:
Event A: "An intact floppy is taken out of the box."

Convenient elementary events for the event A are all intact floppies:
$A = \{i\} \implies |A| = 1$
The number of convenient elementary outcomes is 1.

B.4. Drawing multiple balls with a common property

Theoretical explanation

Let Ω be a space of outcomes with m elements with property ψ and n elements with property ϕ, and let the experiment be the drawing of k balls. For an event A = "all k balls have the property ψ", all elementary events consisting of k balls with property ψ are convenient events, independent of the sequence of the k elements.

Example 1: Drawing balls from an urn

An urn contains one white and three green balls. Two balls are drawn from the urn, each ball is immediately put back into the urn (*drawing with replacement*). The drawn balls have a different colour. Determine the number of convenient elementary events.

Solution:
We denote the possible outcomes of a single draw by "white" and "green", respectively. All pairs of different balls are convenient elementary events. Consequently, we obtain the set

$$A = \{(\text{green}, \text{white}), (\text{white}, \text{green})\}$$

of convenient elementary events. **The number of convenient elementary events is $|\mathbf{A}| = \mathbf{2}$.**

Example 2: Throwing a dice

A dice is thrown twice. Only the numbers 5 and 6 occur. Determine the number of convenient elementary events.

Solution:
Event A: "Only the numbers 5 and 6 occur. "

Convenient elementary events for the event A are the sequences $(5,5)$, $(5,6)$, $(6,5)$, $(6,6)$:
$A = \{(5,5), (5,6), (6,5), (6,6)\} \implies |A| = 4$
The number of convenient elementary events is 4.

B.5. Laplace–probabilities

Theoretical explanation

Theorem by Laplace: If Ω is a finite set of elementary events with an equally distributed probability then the probability $P(A)$ for an arbitrary event $A \subseteq \Omega$ can be computed as the ratio

$$P(A) = \frac{|A|}{|\Omega|} = \frac{\text{Number of convenient elementary events}}{\text{Total number of elementary events}}$$

Random experiments which fulfil both conditions

1. Ω is a finite set of elementary events ω: $\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\}$ with $n < \infty$.
2. All n elementary events $\{\omega_1\}, \{\omega_2\}, \ldots, \{\omega_n\}$ are equally probably, i. e. $P(\{\omega_1\}) = \cdots = P(\{\omega_n\}) = p$.

are called *Laplace experiments.*

Example 1: Drawing balls from an urn

An urn contains four balls in the colours white, red, blue, and green. One ball is drawn from the urn. This ball is white. Compute the probability for this event.

Solution:
Possible elementary events are all four balls: $\Omega = \{\text{white}, \text{red}, \text{blue}, \text{green}\}$. The convenient elementary event is the white ball: $A = \{\text{white}\}$. As a result, we obtain

$$P(A) = \frac{|A|}{|\Omega|} = \frac{1}{4}.$$

Example 2: Taking floppy disks out of a box

A box contains two floppies. One is intact, the other is defective. One floppy is taken out of the box, This floppy is defective. Compute the probability for this event.

Solution:
We have two possible elementary events: $\Omega = \{i, d\}$, and we have one convenient elementary event: $A = \{d\}$. As a result, we obtain

$$P(A) = \frac{|A|}{|\Omega|} = \frac{1}{2}.$$

B.6. Exercise 1

An urn contains five white and five black balls. One ball is drawn from the urn. The drawn ball is black. Compute the probability of this event.

B.7. Events containing elementary events with different properties

Theoretical explanation

Let Ω be a space of outcomes containing m elements with property ψ and n elements with property ϕ. Let the event A contain v ($v < m$) elements of property ψ and w ($w < n$) elements with property ϕ. All sequences of possible outcomes containing v elements of property ψ and w elements of property ϕ are convenient elementary events.

Example 1: Drawing balls from an urn

An urn contains one white and three green balls. Two balls are drawn with replacement from the urn. Exactly one of the balls is green. Determine the number of convenient outcomes.

Solution:
Convenient elementary events are all pairs of balls containing exactly one green ball:

$$A = \{(\text{green}, \text{white}), (\text{white}, \text{green})\} \implies |A| = 2.$$

Consequently, the number of convenient elementary events is 2.

Example 2: Taking floppy disks out of a box

A box contains four floppy disks. Two of the floppies are intact, the other two are defective. Two floppies are taken out of the box with replacement. Exactly one floppy is intact. Determine the number of convenient outcomes.

Solution: All elementary events containing exactly one intact floppy are convenient:

$$A = \{(\text{defective}, \text{intact}), (\text{intact}, \text{defective})\} \implies |A| = 2.$$

Consequently, the number of convenient elementary events is 2.

B.8. Addition of events and probabilities

Theoretical explanation

For two disjoint events A_1, A_2, we obtain

$$P(A_1 \cup A_2) = P(A_1) + P(A_2).$$

The events $A_1, A_2, \ldots, A_n \subseteq \Omega$ are called *pairwise disjoint* if any two different events A_i, A_K ($1 \leq i, k \leq n$ and $i \neq k$) are disjoint. For any sequence $A_1, A_2, \ldots, A_n$ of pairwise independent events, we obtain

$$P(A_1 \cup A_2 \cup \cdots \cup A_n) = P(A_1) + P(A_2) + \cdots + P(A_n).$$

Example: Drawing balls from an urn

An urn contains four balls of the colours white, red, green, and blue. One ball is drawn from the urn. Compute the probability that this ball is red or blue.

Solution:
The space of outcomes is $\Omega = \{\text{white}, \text{red}, \text{green}, \text{blue}\}$. For each of the balls the probability to be drawn is $\frac{1}{|\Omega|} = \frac{1}{4}$. Since the outcomes 'red' and 'blue' exclude each other, the probability of drawing a red or a blue ball can be computed as a sum:

$$P(\{\text{red}\} \cup \{\text{blue}\}) = P(\{\text{red}\}) + P(\{\text{blue}\}) = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$

B.9. Multiplication of events and probabilities

Theoretical explanation

Two events A_1 and A_2 are *(stochastically) independent* if the probability that event A_1 happens does not depend on the occurrence of the event A_2 and vice versa.

For stochastically independent events A_1, A_2 the following *multiplication rule* can be used to compute the combined probability:

$$P(A_1 \cap A_2) = P(A_1) \cdot P(A_2).$$

Example: Drawing balls from an urn

An urn contains three white balls and three green balls. Subsequently, two balls are drawn from the urn and put back. Compute the probability that both balls are green.

Solution:
We regard the event $A =$"both balls are green" as a sequence of two single events $A_1 =$"the first ball is green" and $A_2 =$"the second ball is green". Since the drawn ball is put back before the second ball is drawn, these single events A_1 and A_2 are independent. Therefore, the probability $P(A)$ can be computed as a product of the probabilities $P(A_1)$ and $P(A_2)$. For these single probabilities, we obtain $P(A_1) = P(A_2) = \frac{1}{2}$ and, consequently, for the combined probability, $P(A) = P(A_1) \cdot P(A_2) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.

B.10. Exercise 2

An urn contains five yellow and five black balls. Two balls are drawn from the urn successively. Drawing is performed with replacement. The drawn balls are yellow. Compute the probability of this event.

B.11. Different proportions of properties

Theoretical explanation

If an urn contains balls of different colours in different proportions, the probability of drawing a ball of a specific colour can be computed as

$$P(\textit{Specific colour}) = \frac{\textit{Number of balls with ``specific colour''}}{\textit{Total number of balls in the urn}}.$$

Example: Drawing balls from an urn

An urn contains three white balls and four green balls. One ball is drawn from the urn. Compute the probability that the ball is green.

Solution:
Altogether, there are seven balls in the urn. Consequently, we obtain

$$P(\text{green}) = \frac{\#\text{ green}}{\#\text{ balls}} = \frac{4}{7}.$$

B.12. Exercise 3

An urn contains ten white and six black balls. One ball is drawn from the urn. The drawn ball is white. Compute the probability of this event.

B.13. Exercise 4

An urn contains six black and four white balls. Three balls are drawn from the urn successively. Drawing is performed with replacement. Exactly two of the drawn balls are black. Compute the probability of this event.

B.14. Drawing without replacement

Theoretical explanation

So far, all drawing experiments were performed with replacements. If, in a sequence of drawings, balls already drawn are not put back into the urn before drawing the next ball, the number of balls available in the next drawing is reduced. The single drawings can then be considered to be stochastically independent.

Example: Drawing balls from an urn

In an urn there are three green balls and two white balls. Two balls are successively drawn from the urn *without* replacement. Compute the probability that both balls are green.

Solution:
There is one convenient event: $A = \{(\text{green}, \text{green})\}$. We split up the experiment and the convenient event into two separate experiments and events, respectively. The convenient events then have the form $A_1 = A_2 = \{\text{green}\}$.

In the first drawing, we have five balls in the urn. Three of them are green. We compute the probability of drawing a green ball as

$$P(A_1) = P(\text{green}) = \frac{3}{5}.$$

Afterwards, we have two white balls and two green balls in the urn, i.e. four balls altogether. For the second drawing, we obtain

$$P(A_2) = P(\text{green}) = \frac{2}{4} = \frac{1}{2}.$$

There is no further dependence between the two drawings than the alteration of the numbers of balls. Since this alteration has already been considered, we can regard the two drawings as stochastically independent and, therefore, use the multiplication rule:

$$P(A) = P(\{(\text{green}, \text{green})\}) = P(A_1) \cdot P(A_2) = \frac{3}{5} \cdot \frac{1}{2} = \frac{3}{10}.$$

B.15. Exercise 5

An urn contains five white and five black balls. Two balls are drawn from the urn successively. Drawing is performed without replacement. The drawn balls are black. Compute the probability of this event.

B.16. Generalized descriptions of events

Theoretical explanation

If we have a random experiment which is constructed as a sequence of several single experiments then we have to distinguish between events of the form “exactly k of the n single outcomes have the property Φ” and events of the form “at least (or at most) k of the n single outcomes have the property Φ”. In the latter form, we have to consider the cases of $k, k+1, \ldots, n$ (or $1, 2, \ldots, k$, respectively) convenient single outcomes.

Example: Drawing balls from an urn

An urn contains two white balls and three green balls. Two balls are drawn with replacement. Compute the probability of drawing at least one white ball.

Solution:
All events with at least one white ball are convenient:

$$A = \{\underbrace{(\text{white}, \text{green})}_{A_1}, \underbrace{(\text{white}, \text{white})}_{A_2}, \underbrace{(\text{green}, \text{white})}_{A_3}\}.$$

For the three partial events, we obtain the probabilities

$$\begin{aligned} P(A_1) &= \frac{2}{5} \cdot \frac{3}{5} = \frac{6}{25} \\ P(A_2) &= \frac{2}{5} \cdot \frac{2}{5} = \frac{4}{25} \\ P(A_3) &= \frac{3}{5} \cdot \frac{2}{5} = \frac{6}{25} \end{aligned}$$

Since the three partial events are mutually exclusive, we can compute the final probability $P(A)$ as the sum of the three single probabilities:

$$P(A) = P(A_1) + P(A_2) + P(A_3) = \frac{6}{25} + \frac{4}{25} + \frac{6}{25} = \frac{16}{25}$$

B.17. Exercise 6

An urn contains five black and four white balls. Three balls are drawn from the urn successively. Drawing is performed without replacement. At least two of the drawn balls are white. Compute the probability of this event.