Chapter 7

Case Study: Town Information

7.1 Introduction

In this chapter a case study on the application of the HyperView System is presented. As application domain we chose town information, with an emphasis on cultural information.

We define "Town Information" as information related to events, infrastructure and services of some urban community. In particular we focus on cultural and entertainment related information. Hence, events describe concerts, movie shows, street markets and other more, infrastructure means "locations" of events, i.e., theaters, concert halls as well as restaurants, pubs, bars, etc. Services include public transportation (e.g., bus and subway schedules), ticket reservation, weather forecasting, shopping, and so on.

To a large extent, town information is already available at various Web sites. Although a few servers of city magazines provide ample data on this domain, it is still necessary to consult other servers for additional information. For instance, detailed concert announcements can often only be found at the Web sites of the concert halls or clubs, e.g., of the jazz club "A-Trane"[1] which is presented in this chapter. Also, town maps and public transportation information are typically available from dedicated servers only.

Using the sources mentioned above, it is feasible to plan for instance how to spend an evening in Berlin. However, when trying to find and combine relevant pieces of information, the user is confronted with various problems. The main problem is the *missing link*: related data on different servers are not connected by hyper-links. If the user tries to find related information by consulting another server, he often encounters *semantic heterogeneity* that can make data from one server useless in querying another server. For instance, arbitrary addresses can not be used in the search form of the BVG public transportation server which requires the user to enter names of bus stops.

Another problem is the *incomplete coverage* of single servers. For instance, there is no server that reliably lists *all* concerts in Berlin. To get the full picture, information has to be collected from several Web Sites.

Updates in the sources are another problem. To become aware of changes, the user has to check relevant servers on a regular basis. Finally, new Web sites are set up and existing ones are closed down quite often. This *source fluctuation* makes it necessary to discover new sources and find replacements for those that have stopped operation.

Using the HyperView System to integrate town information from different servers, we can add *missing links* and achieve a more *complete coverage*. The problem of semantically incompatible information in different Web servers cannot be solved by HyperView per se, it only provides the platform to use some conversion service to map information from one Web Site to the terminology required by some other Web Site. Such a conversion service can be some external Web Site, some mapping rules implemented by the HyperView designer, or an internal data base or thesaurus maintained by the HyperView administrator.

The problem of *source updates* can be solved by implementing notification services on top of the database layer of HyperView. Without such a notification service, the HyperView System at least partially automates a search for information and thus makes it easier to repeat it when the information is actually needed.

The problem of data changes is addressed by using the HyperView database primarily as a cache which can be cleared after a source dependent period of time. The rule concept of HyperView aims at making the system robust against small structural changes in the sources and at facilitating the rapid specification of ACR-views over HTML pages.

7.2 Scenario

In this case study we implemented the prototype of an integrated town information server based on the HyperView System. We have included exemplary Web sites that are suitable for demonstrating the potential of such a server and the problems to be solved in practice.

[1]<http://www.a-trane.de/>

The following list presents the exemplary Web sites that have been analyzed for inclusion in the prototype:

- The Web site of the town magazine “Zitty” (<www.zitty.de>) consists of two databases, the “cineProgrammat” for movie theater programs, and the “bplusProgrammat” for all other kinds of cultural events, including concerts, plays, vernisages etc. Both databases offer addresses of locations.

- “Tonwerk” is a music magazine that offers at <www.tonwerk.de> a database of concerts, including occasional links to group / location home pages; its ACR schema is depicted in Figure 7.3.

- The jazz club “A Trane” offers its monthly program at <www.a-trane.de>. The program includes descriptions and pictures of the groups; cf. Figure 7.4

- “Kino Berlin” (<www.kino-berlin.de>) is a service that offers movie theater programs for Berlin. The Film database includes short descriptions and plot summaries. Figure 7.2 shows its ACR schema.

- The “Internet Movie DataBase” (IMDB, <www.imdb.com>) stores detailed information about almost all movies ever produced. For the purpose of a town information service, links to movies and to collections of external reviews are sufficient. Hence the IMDB is not modeled in great detail.

- “Arsenal” is a cineastic movie theater that maintains a Web site with background information on its monthly program (<http://www.fdk-berlin.de/arsenal.htm>).

- The “Stadtplandienst” <www.stadtplandienst.de> offers town maps of Berlin (and other German cities). It allows to lookup a specific address and to present a detail of the town map corresponding to this address.

- Public transportation: at <http://www.fahrinfo-berlin.de> an online schedule of public transportation in the area of Berlin is available. Using HTML forms, the fastest connection between two bus or train stops can be found. However, no search facility for the bus / train stops closest to some address is provided.

The mentioned Web sites can be combined in the following way:

1. cultural events from city magazines (Zitty cineProgrammat and bplusProgrammat), other general event calendars like Tonwerk, and individual promoters (e.g., A-Trane) are merged into a common cultural event calendar

2. detailed background information on events from location Web Sites (e.g. A-Trane, Arsenal) or genre-specific information services (e.g. IMDB) is associated with the events in the calendar

3. links from addresses of locations to town map details are provided

4. links to home pages of locations, artists, ensembles are added (if available)

5. links from events to the public transportation search form are added with *appropriate presettings* for reaching the respective event

7.2.1 Use Case

In the resulting virtual Web site, the task of planning how to spend an evening in Berlin could now be performed as follows:

1. Find interesting entertainment events by either stating preferences in a HTML form or browsing the event database by different categories like service, region, venue, time, etc.
2. Follow links to external pages provided by the event promoters for background information.
3. Follow a link from the address of a venue to a detail of the city map showing its surroundings.
4. Search / browse restaurants and pubs by their category and location
5. Query for a public transportation connection between two locations.

7.3 Developing a cultural event calendar

7.3.1 Conceptual schema

The main idea for the design of the conceptual schema is that cultural *events* are manifestations of cultural *projects*. A cultural project may be the collection of a museum, a concert program in the repertoire of an orchestra or band, the production of a theater play, a movie etc. Accordingly, an event may describe the exhibition of a collection, a particular concert, the performance of a play, or a movie show at a particular movie theater. The cultural events take place at *locations*. Locations in this sense can be museums, concert halls, movie theaters etc.

Since there are many different arts, projects must be organized into different *genre*s such as music, theater, film etc. For practical purposes this classification is still too coarse. Hence the specialization of genres into sub-genres has to be supported. Since this taxonomy must be visible to the user, genres are grouped at instance level into nested collections by their genre. This provides a more flexible data-driven classification scheme than a static a-priori taxonomy.

Since the schema should simplify the navigation for the user, additional edges have to be introduced. For instance, there should be reverse links from events to projects, from projects to genres etc. This can be achieved by simply extending the rules that materialize the forward edges. Furthermore, an event should inherit the location of its project if it is not specified explicitly. Projects belonging to a genre via a *direct_project* edge should belong as well to all more general genres via computed *project* edges. A location should indicate which genres it offers by listing the genres of the projects that have events at that location. These requirements can be satisfied by introducing additional rules that compute these edges.

In Figure 7.1 the resulting schema for the calendar of cultural events is depicted.

7.3.2 Wrapping town information sites

Wrappers for Tonwerk, A-Trane, and Kino Berlin have been implemented. Their schemata are depicted in Figure 7.3, Figure 7.4, and Figure 7.2, respectively. Moreover, a small wrapper for the IMDB that allows to access IMDB records and links to external reviews for a given movie title has been developed as well. The implementation of wrappers for further Web sites can build on this experience.

The Web site of Kino Berlin (see Figure 7.2) offers several indices by which the user can find movies and movie theaters. Most relevant for the event calendar is the index of current movies movie_index that is organized by the initials of movie titles (movie_initial nodes). Each movie is presented on a page (movie_page node) that contains a section with information about the movie

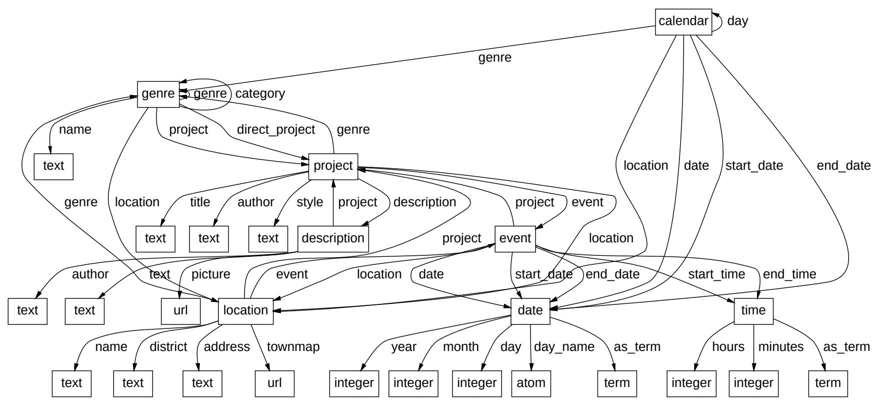


Figure 7.1: DB Schema of a calendar for cultural events.

(movie_info node) and the schedules of the movie at various movie theaters (movie_at_theater node).

The Tonwerk Web site (cf. Figure 7.3) lists concerts in a different calendar for each music style. Each concert is described by the usual attributes, date, time, name of location and band and a short description.

Of the A-Trane Web site, only the program page and its sub-pages (see Figure 7.4) are modeled. This page lists concerts. Since some bands may appear on several consecutive days, a date period is given for each concert. A long description of the band with pictures is on a separate page modeled by atrane_description nodes.

For each wrapper, two rule sets have been implemented: one that extracts data from the HTML code of Web pages and builds the ACR using this data, and one that uses this ACR to build the calendar database conforming to the conceptual schema presented in Figure 7.1.

In this subsection we present only a few small examples of relatively simple rules to give an introduction to the view mechanism of HyperView. We use a graphical notation in which new graph elements added by a rule are shown in boldface. Graph clusters are indicated by dashed boxes with round corners. Lowercase node labels indicate corresponding node types in the schema, while variables are denoted by uppercase labels. More complex rules may involve arbitrary constraints on variables and reuse specifications for reusing existing graph elements.

As first example we present rule atrane_program.concert that extracts information about a concert in the A-Trane. This rule is depicted in Figure 7.5. If a *concert* edge starting from a atrane_program node in the A-Trane ACR is requested, this rule is triggered. It matches a *source* edge pointing to the root node of the HTML page containing the current program of the A-Trane. On this page, it matches rows (tr nodes) of a table listing the concerts. From each row it extracts the text of the cells containing the name of the band and the music style of the band. This information is used to add attributes *group* and *style* to a new atrane_concert node that is connected to the atrane_program node by a new *concert* edge edge.

The information extracted by rule atrane_program.concert is used by rule genre.direct_project to materialize *direct_project* edges for the genre node named jazz. (Since A-Trane is a Jazz club it is assumed that band appearing in this club play Jazz). The rule follows a *atrane_source* edge to the atrane_concert node and the *concert* edge produced by rule atrane_program.concert to match atrane_concert nodes and create for each concert a project node linked to the genre node by a *direct_project* edge. This project node has a *location* edge pointing to the (existing) location node representing the A-Trane. The new *atrane_source* edge pointing to the atrane_concert node is used by other rules to add further data on the concert in the DB layer.

As mentioned above, the calendar graph supports computed *project* edges that point to all projects belonging to a genre *and* to its sub-genres. For instance, all jazz projects belong also to the music genre. In Figure 7.7, rule genre.project that is responsible for materializing *project* edges is

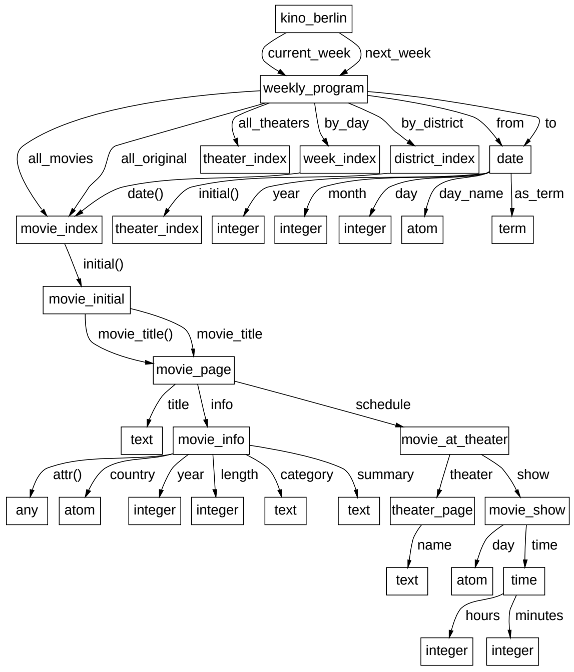


Figure 7.2: ACR Schema of Kino Berlin.

depicted. It consists of two alternative rules, the first one matching all projects belonging directly to a genre and the second one recursively matching all projects of sub-genres.

7.4 The cultural calendar Web site

A prototypical HyperView for integrating the mentioned wrappers into the conceptual schema depicted in Figure 7.1 has been implemented. The generic Web HyperView browser can be used to explore the resulting town information database to be explored. A screen shot of a result page generated by this Web site is shown in Figure 7.8.

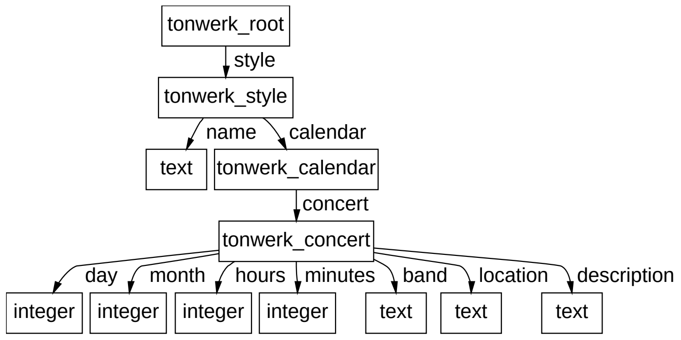


Figure 7.3: ACR Schema of Tonwerk.

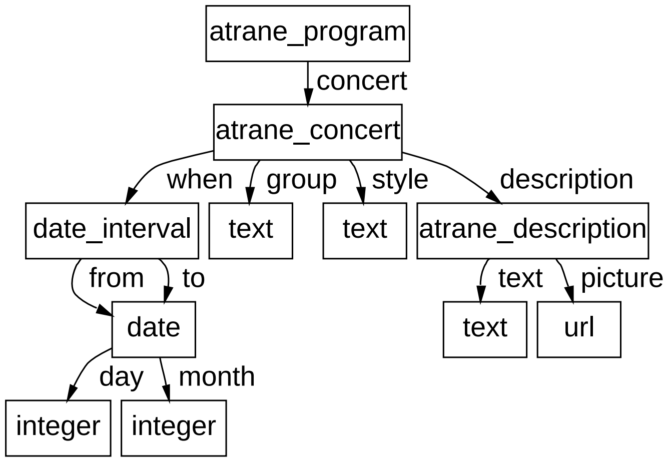


Figure 7.4: ACR Schema of A-Trane.

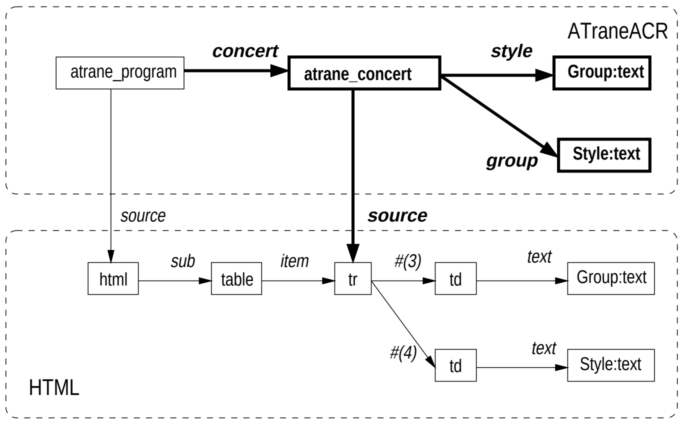


Figure 7.5: Rule atrane_program.concert.

7.5 Summary

The case study presented in this chapter deals with integration of cultural town information. The design and implementation of a HyperView-based prototype for a virtual Web site is described and a screen shot of the resulting virtual Web site is presented. A number of open issues arising from this particular application are identified and discussed. This case study is based on [Faulstich, 1999a].

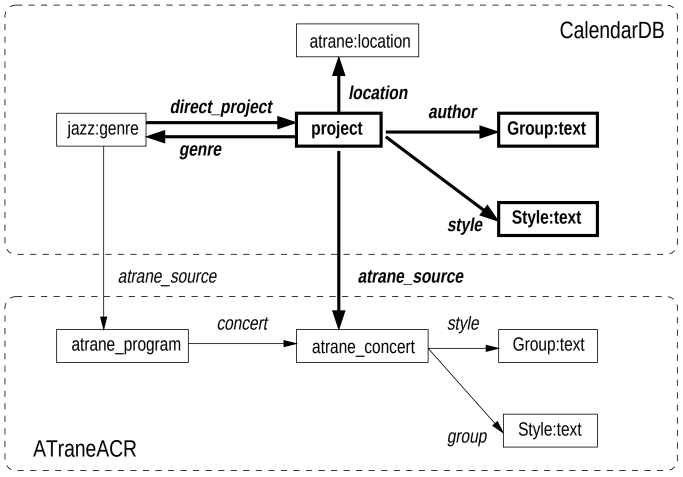


Figure 7.6: rule genre.direct_project.

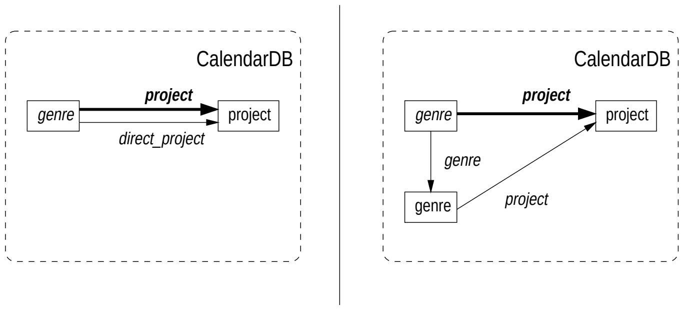
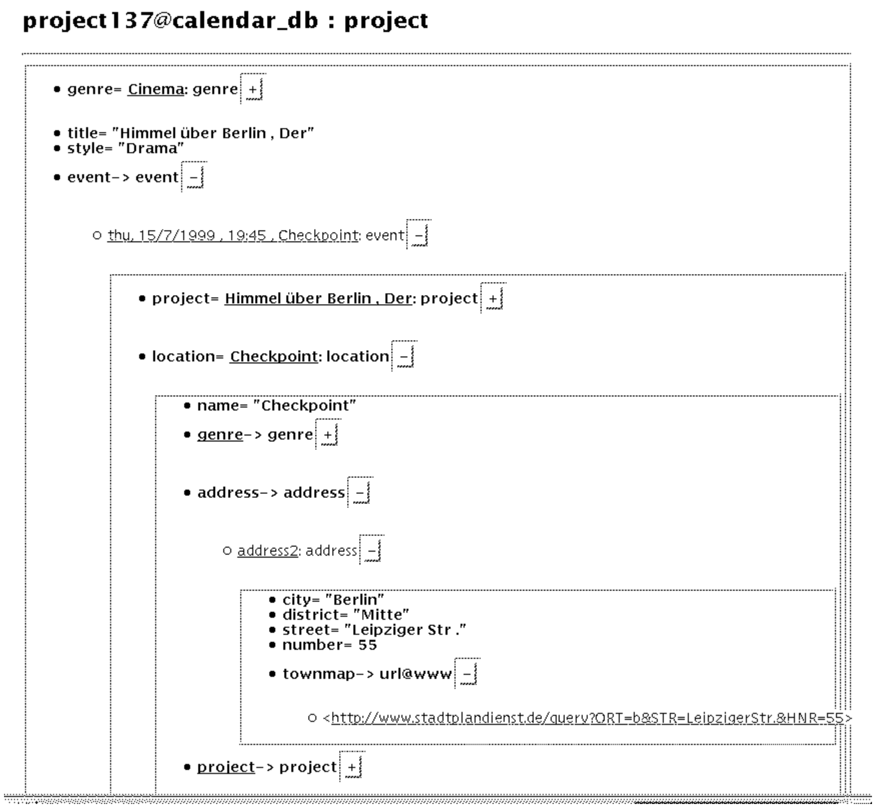


Figure 7.7: Two alternatives of rule genre.project.



project137@calendar_db : project

- genre= Cinema: genre +
- title= "Himmel über Berlin , Der"
- style= "Drama"
- event-> event –
  - thu, 15/7/1999 , 19:45 , Checkpoint: event –
    - project= Himmel über Berlin , Der: project +
    - location= Checkpoint: location –
      - name= "Checkpoint"
      - genre-> genre +
      - address-> address –
        - address2: address –
          - city= "Berlin"
          - district= "Mitte"
          - street= "Leipziger Str ."
          - number= 55
          - townmap-> url@www –
            - <http://www.stadtplandienst.de/query?ORT=b&STR=LeipzigerStr.&HNR=55>
      - project-> project +

Figure 7.8: Screen shot of page describing the film “Wings of Desire” in the cultural event calendar. For the movie, a set of movie shows is presented as “events”, the first of which is expanded. The location of this event, the movie theater “Checkpoint” and its address is also expanded. Note the link from the movie theater’s address to the town map detail at the Stadtplandienst server.