PS Software-Engineering

SE-003-2000 22nd November 2000

Archaeological Data System - ADS

Report

Academic Supervisor Mag. Dipl.-Ing. Schwaiger Roland

Department of Computer Science
University of Salzburg

Correspondence to:

Universität Salzburg
Institut für Computerwissenschaften und Systemanalyse
Jakob–Haringer–Straße 2
A–5020 Salzburg
Austria

Archaeological Data System - ADS

Abstract

This document deals with the development of a software-system with the working title "Archaeological Data System - ADS" for entering, administrating and presenting archaeological data via the internet.

Contents

1 Requirements Definition **3**

1.1 Initial Situation and Objective Target . . . 3
1.1.1 Introduction . . . 4
1.1.2 General Overview . . . 4
1.1.3 Database . . . 5
1.1.4 Entering Data . . . 5
1.1.5 Virtual Museum . . . 6
1.1.6 Additional Notes . . . 7
1.2 System Environment . . . 7
1.2.1 Hardware . . . 7
1.2.2 Software . . . 7
1.2.3 Data . . . 7
1.2.4 Users . . . 7
1.3 User Interfaces . . . 8
1.4 Functional Requests . . . 8
1.4.1 Database . . . 8
1.4.2 Web-Frontend . . . 10
1.4.3 Virtual Museum . . . 11
1.5 Non-Functional Requests . . . 12
1.6 Error Handling . . . 13
1.6.1 Database . . . 13
1.6.2 Web-Frontend . . . 14
1.6.3 Virtual Museum . . . 14
1.7 Documentation Requirements . . . 14
1.8 Acceptance Criteria . . . 15
1.9 Glossary and Index . . . 15
1.10 Risk Analysis . . . 15
1.10.1 Database . . . 15
1.10.2 Web-Frontend . . . 15
1.10.3 Virtual Museum . . . 15
1.10.4 Stability . . . 16
1.10.5 Personnel . . . 16
1.10.6 Hardware . . . 16
1.10.7 Miscellaneous . . . 16

2 Use Case Documentation **18**
2.1 Introduction 18
2.2 System Description 18
2.2.1 System-Level Use Case Diagram 18
2.2.2 Actors 18
2.2.3 Use Cases 20
2.3 Detailed Use Case Description: Primary Use Cases 21
2.3.1 Use Case: Authenticate 21
2.3.2 Use Case: Enter Data 22
2.3.3 Use Case: Enter Scientific Data 24
2.3.4 Use Case: View Data 26
2.3.5 Use Case: View Museum 27
2.3.6 Use Case: Edit Museum 29
2.3.7 Use Case: Search 30
2.3.8 Use Case: CreatePublications 32
2.3.9 Use Case: Administrate 33

Bibliography **34**

Chapter 1

Requirements Definition

Under "requirements definition" we understand the development of contract between our anonymous customer and us, the SE AG. This contract defines exactly, what the software system should be able to do and which premises are valid for the realization.
Since there is no exact definition for "requirements definition" we follow the ANSI/IEEE Guide to Software Requirements Specification [ANSI 830 - 1984] and use the following structure for the requirements definition:

1. Initial Situation and Objective Target
2. System Environment
3. User Interfaces
4. Functional Requests
5. Non-Functional Requests
6. Error Handling
7. Documentation Requirements
8. Acceptance Criteria
9. Glossary and Index

Before the contract between our customer and us can be established, the requirements definition has to be proofed of completeness and correctness and the customer's demands have to be verified whether they can be realized technically and economically. Therefore it is necessary to perform a risk analysis (see Section 1.10 on page 15).

1.1 Initial Situation and Objective Target

This section aims at clarifying topics such as our customer's wishes and requirements concerning the software, it shall give restrictions what the software is not to do as well as point out questions which are to be discussed with our customer.

As R. Yeh and P. Zave describe [1], a great deal of time and money is invested in maintenance of a software-product. To reduce this in order to be more productive as you can

concentrate on other projects, a very important part of software-developement is to specify exactly the system's requirements.
Yeh and Zave also mention 3 major steps which should be persued during this phase of the software-project:

1. *Problem identification:* One shall identify and describe the needs of a system for certain purposes;
2. *Problem understanding:* Collect and analyze information about the system and its environment, as well as their interaction;
3. *Problem specification:* Describe the behavior of the system.

This section is the product of these 3 steps.

1.1.1 Introduction

Our company was engaged to develop a software-system with the working title "Archaeological Data System - ADS" which first of all shall:

- Provide an interface via the world-wide-web enabling our customer to enter archaeological data such as tombs, settlements or artefacts into a central database.
- Enable third parties to query the database for information via the world-wide-web.
- Present the data also via a virtual museum.

We are responsible for the design and implementation of this system including the following points:

- Web-front-end to manipulate and display data.
- The design and implementation of the database itself.
- Maintain these services.
- Host the database on a server or at least administrate this work.

1.1.2 General Overview

As the database shall be accessible via a standard web-browser, we will not develop an independent program which could connect to the database for example via TCP/IP. Just the web-frontend for web-browsers will be designed and implemented.

Our Customer has no experience with a similar software, so we have to organize seminars to teach the customer how to operate the system and additionally write a documentation manual for the software. This is necessary because once ADS is in action the customer himself is responsible for the system. We only will act as ASP (Application Service Provider) then.

There are no limitations in the financial capital we have for this project. In January 2001 the Inception- and Elaboration-Phase of the project should be finished and maybe we can present our customer a prototype.

1.1.3 Database

Currently there are 4000 records to enter into the database. A record will consist of approximately 20 attributes and can also contain pictures, sketches, movies and sounds. The expected annual growth of data is about 400 records at all (400 pics, 400 sketches, 10 movies and several sound-samples). The multimedia-files have to be uploaded thru our web-interface. These files will not be stored into the database, but rather using the filesystem. The database will only store the path to the respective files.

The attributes for the different fields can be found in the next chapter.

To search in the database, our web-frontend will provide a search-screen. There will be three kinds of queries:

Simple query: Just enter a keyword and all the attributes will be compared to it.

Extended query: One can specify, which data certain attributes should contain and connect all the search-options by logical operators.

Query in the museum: The visitor can search for special items in the museum.

The data collectors can specify certain attributes of a record, the visitor cannot see when searching in the database. He can also make whole records invisible for the visitors.

As the customer wishes the option to have several virtual museums at once, one of the records attributes will propably be, to which virtual museum a specific object belongs to.

1.1.4 Entering Data

There are two main groups who can enter data into the database:

1. Data collectors
2. Scientific data collectors

Members of the first group will enter general data concerning the archaeological object such as the place where it was found, while members of the second group will add additional information such as interpretations and publications concerning this object.

To address all questions which have to do with "Who has the right to do what in the database?" a database-access-system has to be developed. It should enable the customer to organize the users of the software-system in groups and equip them with certain rights (read, write, modify).
So ADS will provide a user-management such as known from UNIX. There will be four groups so far:

Data collectors: They can enter and manipulate general data concerning the archaeological objects. They also can define relationships between different types of data or specific records. Data collectors also can edit the virtual museum where the objects are to be presented.

Scientific data collectors: They can also enter and manipulate data (even more specific one, such as publications concerning the archaeological objects). Again they have the option of defining relationships as mentioned before as well as editing the virtual museum. So this group is something like an extension of the data collectors.

Visitors: They will just have the option to query the database for information as well as to walk through the virtual museum.

Administrators: The administrators will be able to do anything they want within the system, so they have the ability to maintain and repair small but urgent concerns, perhaps via the internet.

In the design of ADS we will keep an eye on the ability of adding new groups or changing groups in future versions.

1.1.5 Virtual Museum

The software-system shall include a virtual museum where the visitors can walk through and zoom into descriptions of archaeological objects and artefacts. The museum shall not host all the data collected as the customer wants to perform special exhibitions on current topics.
There is no need to model the museum with 3D-graphics. Nevertheless, we have several options:

- Show photography-like pictures of the rooms, move within the museum by clicking on arrows or signs.
- Present the museum animated (via tools such as Quicktime, Flash, Shockwave, etc.)
- Just show a clickable room-map.
- Others.

At this point we want to remind the customer of the fact that the virtual museum will be accessed via the internet and as the majority of visitors will have low-speed connections, they will not be able to fully enjoy the museum if there is too much (graphical) data to be transferred.

The museum will have three floors, each of which consist of about ten rooms. In each room there are several showcases which contain archeological artefacts. The data collectors have the ability to design the museums as they like (so the scientific data collectors also have). And of course the administrator can also manipulate the museums.
For designing museums, it will be best to either offer a given set of different showcases, frames and boards to choose from or to give a list, where one can choose which entries of the data-record will be shown.

The museums are usually used for special exhibitions, so there will not be all the items from the database in the museum. The visitors have the ability to search here for archeoligical artefacts. As a result they will get a discritpion how to get to the desired item in the museum. If one selects a item in a showcase he is given the available data from the database.

1.1.6 Additional Notes

The customer wants to produce represantative CDs as well as catalogues of the virtual museum and its exhibitions. These CDs will only contain the museum itself (accesible thru a web-browser) and the corresponding extract from the database. The customer himself is responsible for the production of this CD. So we will add a feature which automatically copys the museum-data in a certain directory, so that he can burn the contents of this directory on a CD-R. An equivalent feature will exist to create the catalogue.

Our management demands the use of XML in order to be as platform-independent as possible.

1.2 System Environment

1.2.1 Hardware

As mentioned above, ADS will be hosted by our company. We will be responsible for readiness for use and maintenance of the system and also accessibility via the internet. Therefore we need a web-server, on which we run ADS. Further we need a database-server, to implement the database for ADS. We shall use the latest hardware from a well-known and reliable vendor.
All the ADS-user needs, is a computer with connection to the internet (e.g. via modem)

1.2.2 Software

For our servers (web and database) we need a reliable and stable operating system (such as Solaris, HP-UX, Linux) and web- and database-server-software to run under this OS.
The demand of software for the ADS-users is very small. Only an OS and a web-browser is required.

1.2.3 Data

The database will initially contain about 4000 records (each has about 20 attributes) of archeological data. Every year there will be about 400 new records to enter the database. Since the customer himself is responsible to fill the database we don't need to know anymore.

1.2.4 Users

The system distinguishes between four user-groups. Three active groups (data collectors, scientific data collectors and administrators) who can change the database and one passive group (visitors) who can only query the database. The visitors will be the ones accessing ADS most of the time (maybe 50-500 visitors per day), while (scientific) data collectors will use it approximately twice or three times per day (400 new entries per year plus manipulations of existing data). The administrator is expected to work very few with the system.

1.3 User Interfaces

This section will come up within the next assignment.

1.4 Functional Requests

The functional requests in this section are formulated using the formal language which is systematically described thru the Backus-Naur-Form as the following:

- <Request> ::= <Request Name>: <Request Text>. Request Type: <Request Type>. Priority: <Priority>. Reason: <Reason>
- <Request Name> ::= <Text>
- <Request Type> ::= HAVE TO|SHOULD|MUST NOT
- <Priority> ::= 1|2|3|4|5|6|7|8|9|10
- <Reason> ::= <Text>
- <Text> ::= <Character><Text>|<Character>
- <Character> ::= a|...|z|A|...|Z|0|...|9|"|.|...|;

Priorities are ranked 1 for highest and 10 for lowest priority.

1.4.1 Database

- DB1: The database has to be able to handle the initial 4000 records plus 400 new records per year over several decades (every record consists of about 20 attributes, containing text and maybe multimedia-files)
- Priority: 1
- Reason: This is a crucial part of the software-system.

- DB2: Individual attributes of the data-records have to be lockable and whole data-records have to be hideable.
- Priority: 2
- Reason: Some attributes as well as whole records must not be search-/viewable by the visitors.

- DB3: The collected data has to be editable.
- Priority: 1
- Reason: One should have the possibility to edit already collected data in order to add or correct informations.

- DB4: The System has to distinguish between different user-groups.
- Priority: 1
- Reason: This is necessary to deal with the rights the different users have.

- DB5: There has to be a group for each of the following: administrators, data collectors, scientific data collectors and visitors.
- Priority: 1
- Reason: The administrators are necessary to maintain and repair urgent concerns. The other three groups are demanded by the customer.

- DB6: The group-management has to be extendable.
- Priority: 4
- Reason: For future releases or updates this can be very useful and time-saving.

- DB7: An artefact record will contain the following attributes
  - inventory-number
  - name
  - category
  - scientific text
  - size
  - material
  - the name of the person who found the artefact
  - picture
  - video
  - sound
- Priority: 1
- Reason: This is are the essential attributes demanded by the customer.

- DB8: A grave-record will contain the following attributes
  - gps-data
  - gis-data
  - scientific text
  - the name of the person who found the grave
  - picture

  - video
  - sound

- Priority: 1
- Reason: This is are the essential attributes demanded by the customer.

- DB8: A settlement-record will contain the following attributes
  - gps-data
  - gis-data
  - scientific text
  - the name of the person who found the settlement
  - picture
  - video
  - sound
- Priority: 1
- Reason: This is are the essential attributes demanded by the customer.

- DB9: There will be a 1..n relationship between graves and artefacts. Thus, n artefacts can be associated with a grave.
- Priority: 1
- Reason: It is essential to link graves to artefacts.

- DB9: There will be a 1..n relationship between settlements and artefacts. Thus, n artefacts can be associated with a settlement.
- Priority: 1
- Reason: It is essential to link settlements to artefacts.

1.4.2 Web-Frontend

- WF1: There has to be a seperate section for the (scientific) data collectors which has to be protected by a password system.
- Priority: 1
- Reason: The section where the (scientific) data collectors can edit the database has to be seperated and protected from the visitors.

- WF2: The (scientific) data collectors have to be able to enter data in form of text into the databank, as well as upload images, audio- and videofiles.

- Priority: 1
- Reason: One of the crucial functions of the software-system is to allow the (scientific) data collectors to update the database via the web-frontend.

- WF3: There has to be a simple text-search function, which searches all the records and their attributes which contain text (except hidden records and locked attributes) for the entered search word.
- Priority: 1
- Reason: The visitor has to be able to search the database in a simple way.

- WF4: There has to be an advanced search function, which searches for specific attributes (except hidden records and locked attributes). The entries have to be connected with logical operators.
- Priority: 3
- Reason: The advanced visitor has to be able to search the database more specifically.

- WF5: There has to be a searchable knowledge tree which leads to the items.
- Priority: 5
- Reason: The visitor should be able to start his search in a general field of interest and specify his search as he moves up the knowledge tree.

- WF6: The web-frontend must not be based on additional plug-ins for web-browsers.
- Priority: 1
- Every visitor shall have the same access to our system without downloading and installing plug-ins.

1.4.3 Virtual Museum

- VM1: There has to be a Virtual Museum with initially 3 floors and 10 rooms per floor (there are several showcases in a room, where certain artefacts from the database are on display).
- Priority: 1
- Reason: This is a crucial part of the software-system. The customer has to be able to display certain artefacts.

- VM2: The visitors have to be able to browse through the museum.

- Priority: 1
- Reason: The visitors must be able to view the items that are on display in the showcases.
- VM3: The Virtual Museum has to be expandable and editable.
- Priority: 3
- Reason: The customer wants to edit the layout of the rooms and showcases as well as add additional floors to the museum.

- VM4: There has to be a search function in the virtual museum.
- Priority: 3
- Reason: The visitor might like have a description how to get to the object of his interest.

- VM5: The museum has to be able to host an exhibition concerning a special topic.
- Priority: 4
- Reason: This is a feature our customer demands.

- VM6: The museum has to be depicted by sophisticated graphics.
- Priority: 7
- Reason: Visitors should get an comfortable and eye-catching surface.

- VM7: The (scientific) data collectors has to be able to create a representing CD-ROMs and printed catalogues of a special exhibition in the virtual museum.
- Priority: 2
- Reason: This is a feature our customer demands.

1.5 Non-Functional Requests

Factors such as the reaction time of our software-system depend very much on third-party's performances or other factors (e.g. internet-providers, rush-hours on the data-highway). In combination with the fact that our company uses the latest hardware and of course competent and efficient programming, it therefore makes little sense to give exact statements such as "The database gives an answer within 2.5 seconds". If the customer wants the final product to be faster than the prototype, we will try to do some improvements of performance if possible.

The stability of the system shall be reached by using high-quality hardware and a reliable operating system. Of course stability and usability also depends on the quality of our product.

The use of XML combined with the fact that no special plug-ins but just a plain web-browser are going to be used grant high portability. This is necessary in order to offer features such as producing CDs or catalogues.
XML will be an intermediary layer in the system, enabling us to publish content in HTML, PDF, or what ever format wished for by the customer.

1.6 Error Handling

1.6.1 Database

- DB-ERR1: The database doesn't react to queries from the web-frontend.
- Cause: The DB-server-daemon failed or isn't loaded.
- Solution: Generate an error message and inform the technical administrator.

- DB-ERR2: A query doesn't return any results although the desired information is definitely stored in the database.
- Cause: This might be due to malformed DB-queries.
- Solution: Our company has to go thru these routines and fix the error.

- DB-ERR3: The database gives too much information (e.g. shows records or attributes which are actually locked).
- Cause: This might be due to malformed DB-queries.
- Solution: Our company has to go thru these routines and fix the error.

- DB-ERR4: The (scientific) data collectors can't add or edit any information.
- Cause: The user is assigned to a wrong group or the rights of that group are not set correctly. The file-system gives a write-permission-error. The system resources are too low (e.g. not enough free disk-space).
- Solutions: The administrator has to assign the user to the right group or set the group rights correctly. The administrator has to check and correct the environment of the system.

1.6.2 Web-Frontend

- WF-ERR1: The web-frontend does not load.
- Cause: The hardware failed. The web-server-daemon failed. Third-party failure (such as the user's provider can't connect to our system).
- Solution: Contact the technical administrator.

- WF-ERR2: The web-frontend does not load properly. Parts of the page are missing.
- Cause: Files are missing or not set up correctly.
- Solution: Contact the webmaster.

1.6.3 Virtual Museum

- VM-ERR1: The Virtual Museum (or single parts of it) does not load.
- Cause: The server is down. Several files are missing.
- Solution: Contact the technical administrator.

1.7 Documentation Requirements

As our customer has no previous experience with similar software-systems and wishes to be independent after finishing the project, the product manual has to contain detailed instructions dealing with the following issues:

- Modification of the data-access-system. Explains how to
  - change the rights of certain groups
  - add users and groups
  - change the group of a user
- Modification of the Virtual Museum. Explains how to
  - add floors, rooms, showcases and artefacts to the museum
  - equip rooms and showcases with topics
  - change the layout of rooms and showcases
- Update the database. Explains how to
  - enter data-records into the database
  - upload files into the database

Because ADS isn't a commercial software-product and the customer didn't demand it, there's no use to make a printed handbook. The documentation will be only available as a Software-Document (such as Postscript, Adobe Acrobat, HTML, ...) from which one can make a hardcopy.

1.8 Acceptance Criteria

The most important criteria which have to be fulfilled so that our customer accepts our product are listed in the sections above. If we can deliver the product in time while implementing the reqired features, this should lead to a positive acceptance.

1.9 Glossary and Index

Data-Access-System: An administrative concept to handle the rights of groups of users, defining their rights to read/write/modify data.

Knowledge Tree: A search method which starts with a general topic. The user can then choose between several branches and in doing so specify the topic he is interested in.

Router: A device use to interconnect two or more LAN's together, each of which operates with a different medium access control method - also known as a gateway or intermediate system.

TCP/IP: The complete suite of protocols including IP, TCP and the associated application protocols.

1.10 Risk Analysis

As an importanat part of the cost-benefit calculations we're performing a risk analysis for ADS. According to E. Yourdon [2] risk analysis should always be done, even if nobody asked for it. This is because it can be helpful to know which things can go wrong and thereby can be the cause for a lot of problems and additional costs.

1.10.1 Database

The expected amount of data in the database is rather modest and should not be a problem for a modern database management system. If the system should run short of storage space in the future, it will be no problem to provide additional disk space. Hence, the risk of problems with that part of the software-system can be classified as low.

1.10.2 Web-Frontend

The only risk connected with the web-frontend is, that the customer might not be satisfied with our design proposals. It will be important to show him some basic layouts in an early stage, to make clear what the customer is looking for and in doing so, avoid the risk of wasting too much time and manpower on unnecessary design work. The technical specifications of the web-frontend should not be a problem for our experienced developers.

1.10.3 Virtual Museum

Concerning the virtual museum, it has to be said that the customer has very loose ideas of what it should finally look like. This might lead to problems, if the customer is not satisfied

with our developer's suggestions. To avoid the necessity of last-minute adjustments of the whole concept, this risk should be minimized by clarifying if our proposals meet his demands.

Another risk lies in the fact that the museum has to be viewable without the aid of additional plug-ins. On the one hand this might result in an increased need of time and work to find solutions for otherwise easily solveable problems (e.g. when using Flash- or Quicktime-plug-ins), on the other hand it minimizes the risk of compatibility problems.

1.10.4 Stability

As the system shall always be up and we are responsible for it's stability and readiness for use, this point is quite important to ensure smooth working. We therefore should use a well-known and reliable hardware and operating system for the servers as well as make a contract with a service-company for case that the hardware fails. The use of XML and possible choice of SQL as the basis for database lead to a quite platform-independent design of the core-components.

1.10.5 Personnel

As our project team is rather small, it can have a great influence on the software-development and on the project if somebody leaves the team. If we had to recruit new personnel this could cost a lot of time and money.
On the other side the facts, that within our team the project-leader changes with every new assignment and every team member has to work out an individual solution for the current assignment, have the effect that every team-member is on the same know-how-level. So, if one team-member will be disabled for just a little time, this will not handicap the whole team. But if three or more members drop out for the same time, the project is endangered.

1.10.6 Hardware

The only limitation to the used hardware is it's stability. If it turns out that there are not enough resources, such problems can easily be solved by adding more resources. As we have no limitations concerning financial investments, the hardware shouldn't be a problem.

1.10.7 Miscellaneous

Concerning some risks mentioned by E. Yourdon [2] and S. Vollmann [3] the following can be said:

- If there is much traffic on the internet, the response time may drop down. Since we can do nothing about this, we have to accept the risk of partially slow connections.
- In case of electric power breakdown, our power generator will start up, to ensure power supply for our servers. Customers won't recognice that at all.
- It remains negotiated with our customer who shall be held responsible for possible loss of data. Nevertheless, such a risk shall be minimized by making regular backups.

- It is the responsibility of our management to evaluate our cutomer's credit-worthiness.

Chapter 2

Use Case Documentation

2.1 Introduction

As [4] states, when defining primary uses cases, it is not needed to capture "every detail of the use cases". Thus, use cases are used primarily to capture the high level user-functional requirements of a system (cf. [5])
Keeping this definition in mind, we design our use-case-model not primarily as a design-concept, but rather as an auxiliary means for communicating with the customer. The primary use-cases reflect the basic functionality of the ADS. Of course, the model is abstract and these primary use-case are constituted by a lot of sub-use-cases (e.g. Administrate is a very abstract use-case, consituting of CreateBackup, ManageRights, etc).

2.2 System Description

2.2.1 System-Level Use Case Diagram

see Figure 2.1 on page 19

2.2.2 Actors

- InternetUser

  The InternetUser is interested in viewing the data and museums. He ist the one visiting the webpage of the ADS and using the ADS for display-purposes

- Designer

  A Designer creates Virtual Museums. He can arrange floors, room and vitrinas, but is not necessarily able to enter data. Furthermore, a Designer is able to create publications, such as a CD-Rom or a catalogue.

- DataCollector

  A DataCollector is able to enter data concerning artefacts, graves and settlements into the system. DataCollectors (as they are generalized by Designers) can create and modify virtual museums.

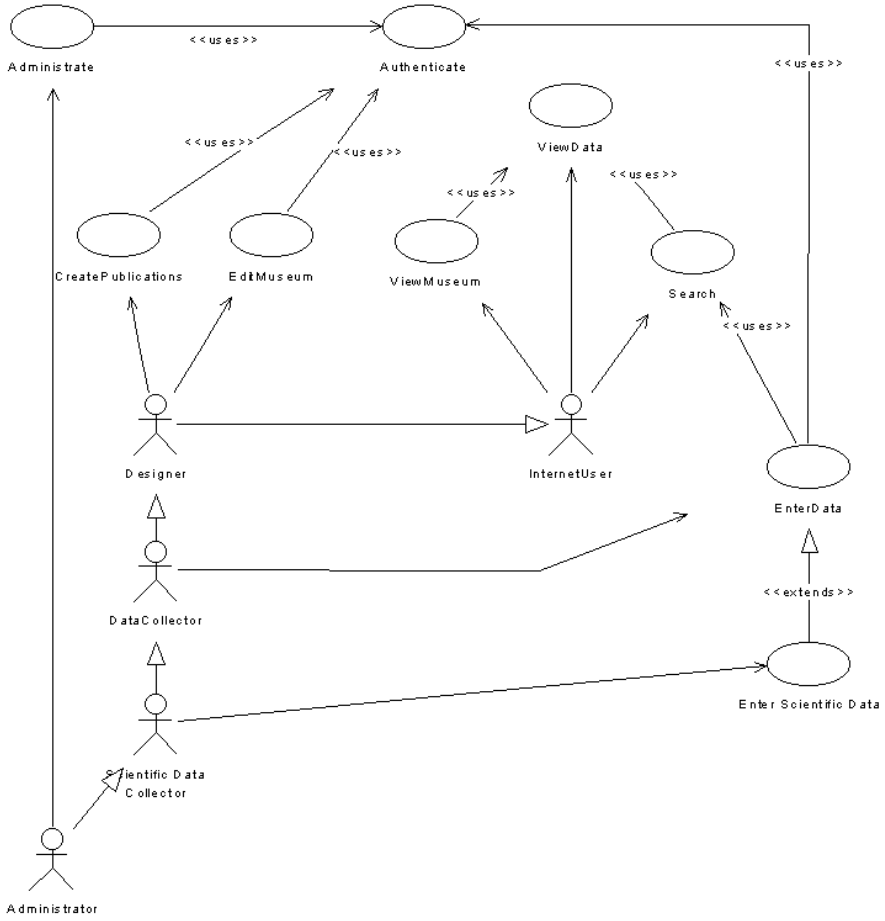

Figure 2.1: System-Level Use Case Diagram

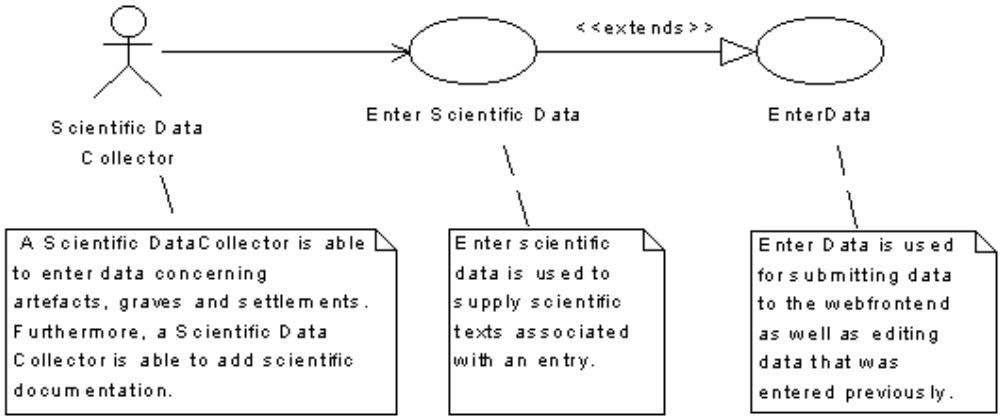

Figure 2.2: Actor: Scientific Data Collector, Use Case: Enter Scientific Data

- Scientific DataCollector

  A Scientific DataCollector is able to enter data concerning artefacts, graves and settlements. Furthermore, a Scientific DataCollector is able to add scientific documentation. Scientific Data Collectors (as they can generalized by DataCollectors) can create and modify Virtual Museums.
  For reference purposes the Use-Case: Enter Scientific Data has been added for this actor - see Figure 2.2 on page 20)

- Administrator

  An administrator has all the rights of the other groups, i.e. he is able to enter data, scientific, etc. Furthermore he has the privilege to edit rights and groups.

2.2.3 Use Cases

The use-cases will be explained in more detail in section 2.3.

- Authenticate
- View Data
- Enter Data
- Enter Scientific Data
- Search
- View Museum
- Edit Museum
- CreatePublications
- Administrate

2.3 Detailed Use Case Description: Primary Use Cases

2.3.1 Use Case: Authenticate

Authenticate is used to identify a user and associate him with a group with more rights than an InternetUser, i.e. DataCollector, Scientific DataCollector, Administrator.

Actors

- DataCollector
- Scientific DataCollector
- Administrator

Priority

Priority: 3
Reason: The authentication can be implemented after other use-ases have been done.

Preconditions

- "Login" pressed

Postconditions

- The respective actor is authenticated as DataCollector, Scientific DataCollector or Administrator or authentication fails.

Extension Points

none

"Used" Use Cases

none

Flow of Events

Primary Scenario

1. Actor selects "Login"
2. Actor supplies Login-name
3. Actor supplies Password
4. Login-Name and Password match
5. User-Data (and associated group) is retrieved from Data-Base

6. User is logged in and granted rights according to his group-association

Secondary Scenario

1. Actor selects "Login"
2. Actor supplies Login-name
3. Actor supplies Password
4. Login-Name and Password do not match
5. System displays an error-message
6. User is not logged in

User Interface

- Input-field for login-name
- Input-field for password (hidden input!)
- submit-button

Subordinate Use Cases

none

Other Requirements

none

2.3.2 Use Case: Enter Data

Enter Data is used for submitting data to the webfrontend as well as editing data that was entered previously.

Actors

- Data Collectors (and thus also Scientific DataCollectors as they are generalized by DataCollectors, and Administrators as they are generalized by Scientific DataCollectors) [1]

Priority

Priority: 1 Reason: Entering Data is an essential part of the system.

[1] In the description of further use cases, if not stated otherwise, it will be presumed that if an actor is generalizes by another actor, the former inherits the rights of the latter. (Cf. Figure 2.1 on page 19 for "generalize"-hierarchy of actors)

Preconditions

- "Enter Data" is selected by an actor with the rights to do so

Postconditions

- Data is stored in the database or use-case fails

Extension Points

none

"Used" Use Cases

Use Case Authentication

Flow of Events

Primary Scenario

1. Actor selects "Enter Data"
2. Actor has the right to enter data
3. Actor enters data into user-interface
4. Actor presses submit-button
5. System submits data to the database
6. Data is stored
7. A confirmation is displayed

Secondary Scenario A

1. Actor selects "Enter Data"
2. Actor has the right to enter data
3. Actor enters data and paths to multimedia data into user-interface
4. Actor presses submit-button
5. System submits data to the database
6. System uploads multimedia files and stores them using the filesystem.
7. Data is stored, paths to multimedia files are stored
8. A confirmation is displayed

Secondary Scenario B

1. Actor selects "Enter Data"
2. Actor selects "Edit Data"
3. Actor has the right to edit data
4. Actor selects the entry to modify
5. System displays data entered in the database
6. Actor changes data
7. Actor presses submit
8. Data is changed in Database
9. A confirmation is displayed

Secondary Scenario C

1. Actor selects "Enter Data"
2. Actor does not have the right to enter data
3. An error message is displayed

User Interface

- input-fields to supply textual information for each textual attribute
- input-fields to supply paths to multimedia-files for eacht multimedia-attribute
- submit-button

Subordinate Use Cases

none

Other Requirements

none

2.3.3 Use Case: Enter Scientific Data

Enter scientific data is used to supply scientific texts associated with an entry.

Actors

- Scientific Data Collectors

Priority

Priority: 1 Reason: Entering Data is an essential part of the system.

Preconditions

- "Enter Scientific Data" is selected by an actor with the rights to do so

Postconditions

- Data is stored in the database or use-case fails

Extension Points

none

"Used" Use Cases

Enter Data

Flow of Events

Primary Scenario

1. Actor selects "Enter Scientific Data"
2. Actor has the right to enter scientific data
3. Actor enters data into user-interface
4. Actor presses submit-button
5. System submits data to the database
6. Data is stored
7. A confirmation is displayed

Secondary Scenario A

1. Actor selects "Edit Scientific Data"
2. Actor has the right to edit data
3. Actor selects the entry to modify
4. System displays data entered in the database
5. Actor changes data
6. Actor presses submit

7. Data is changed in Database
8. A confirmation is displayed

Secondary Scenario B

1. Actor selects "Enter Scientific Data"
2. Actor does not have the right to enter scientific data
3. An error message is displayed

User Interface

- input-fields to supply textual information for each textual attribute
- submit-button

Subordinate Use Cases

none

Other Requirements

none

2.3.4 Use Case: View Data

The actor in this use case is interested in displaying a single item.

Actors

- InternetUser

Priority

Priority: 1 Reason: Displaying data actually is one of the main purposes of ADS

Preconditions

- an item is selected by the actor (e.g. through Search or ViewMuseum)

Postconditions

- item-data or an error-message is displayed

Extension Points

none

"Used" Use Cases

Authenticate

Flow of Events

Primary Scenario

1. Actor selects an item
2. System checks users rights
3. Systemretrieves data fields of that item, the user is entitled to see, from the database
4. Data is displayed in appropriate from

Secondary Scenario

1. Actor selects an item
2. System cannot find item in database
3. An error-message is displayed

User Interface

see storyboard

Other Requirements

none

2.3.5 Use Case: View Museum

This use case refers to the process of an actor viewing a museum. The actor is able to navigate through floors, enter rooms, look at vitrinas and select items from a vitrina for more detailled display.

Actors

- InternetUser

Priority

Priority: 4 Reason: Museums will be implemented after several other use-cases have been implemented.

Preconditions

- actor selects "View Museum"

Postconditions

- Museum or an error-message is displayed

Extension Points

none

"Used" Use Cases

View Data (and thus also Authenticate)

Flow of Events

Primary Scenario

1. Actor clicks on "View Museum"
2. System checks users rights
3. System displays museum
4. Actor can naviage through the museum (i.e. floors and rooms) and/or view single items or vitrinas

Secondary Scenario

1. Actor selects "View Museum"
2. No museum is active
3. An error-message is displayed

User Interface

- display of museum
- navigation
- display of rooms
- display of vitrinas
- display of artefacts

Other Requirements

none

2.3.6 Use Case: Edit Museum

Actors

- DataCollector

Priority

Priority: 3

Preconditions

- "Edit Museum" is clicked by the actor
- Actor has the right to edit museums

Postconditions

- Museum data is edited or an error-message is displayed

Extension Points

none

"Used" Use Cases

Authenticate

Flow of Events

Primary Scenario

1. Actor selects a museum (if there are several)
2. System checks actor's rights
3. Actor has the right to edit museum
4. Museum data is displayed
5. Actor modifies data (i.e. creates/modifies rooms, vitrinas, or delete museum)
6. Actor presses "submit"
7. System updates Museum

Secondary Scenario A

1. System checks actor's rights
2. Actor has the right to edit museum

3. No museum is established yet
4. System adds museum
5. Flow of events continues as from step 4. in primary scenario

Secondary Scenario B

1. Actor selects a museum (if there are several)
2. System checks actor's rights
3. Actor does not have the right to edit museum
4. An error-message is displayed

User Interface

see storyboard

Subordinate Use Cases

AddMuseum
N.B. This use cases will be documented in a later stage when we are dealing with use-cases in more detail.

Other Requirements

none

2.3.7 Use Case: Search

The actor searches the whole data he is entitled to see or searches a museum for specific entries. There are two kinds of searches: basic search and complex search.

Actors

- InternetUser

Priority

Priority: 1
Reason: Searching the museum will make navigation a lot easier for the users

Preconditions

- Actor selects "Search"

Postconditions

- System displays search results or an error-message

Extension Points

none

"Used" Use Cases

ViewData

Flow of Events

Primary Scenario

1. Actor selects "Search all data"
2. Actor enters query (simple or complex)
3. System checks actor's rights
4. System searches data according to user's rights
5. System displays a list of found items
6. Actor selects an item
7. System displays that item ($\rightarrow$ Use-Case: ViewData)

Secondary Scenario

1. Actor is in a museum
2. Actor selects "Search museum"
3. Actor enters query (simple or complex)
4. System checks actor's rights
5. System searches data according to user's rights
6. System displays a list of found items
7. Actor selects an item
8. System displays that item ($\rightarrow$ Use-Case: ViewData)

In both of the above scenarios, at step 5. if no items are found, an appropriate message is returned.

User Interface

- A possibility to choose between basic and complex search
- Masks for entering the query for both basic and complex search
- Display of results

Subordinate Use Cases

SearchData, SearchMuseum
N.B. These use cases will be documented in a later stage when we are dealing with use-cases in more detail.

Other Requirements

none

2.3.8 Use Case: CreatePublications

The actor selects a museum or a part of the data in order to publish it in a catalogue or a CD-Rom.

Actors

- Designer

Priority

Priority: 5
Reason: The creation of publications can be easily added after the main system is designed.

Preconditions

- "Publish" pressed

Extension Points

none

"Used" Use Cases

Authenticate

Flow of Events

Primary Scenario

1. Actor selects "Publish Museum"

2. If there are several museums, the system displays a list of museums.
3. If there are several museums, actor chooses one
4. Actor chooses way of publishing
5. System publishes the data in the specified way

Secondary Scenario

1. Actor selects "Publish Data"
2. System displays entries the actor is entitled to view
3. Actor choosed data to publish
4. Actor chooses way of publishing
5. System publishes the data in the specified way

User Interface

see storyboard

Subordinate Use Cases

none yet

Other Requirements

none

2.3.9 Use Case: Administrate

This use-case deals with the various tasks that need to be performed by an administrator, such as maintanence of the syste, creating backups, managing groups, managing rights, etc.
It will evolve during the implementation, thus the description here is not meant to be taken as complete.

Actors

- Administrator

Priority

Priority: 1

Preconditions

- Actor logs in as administrator

Postconditions

none

"Used" Use Cases

none yet

Flow of Events

As this use-case will be made up of a lot of differen sub-ordinate use-cases a flow of event will not be given at this point.

User Interface

see storyboard

Subordinate Use Cases

ManageRights, ManageGroups, CreateBackup, etc. (This list is by no means complete!)

Other Requirements

- login as administrator has to be secure

Bibliography

[1] R. T. Yeh and P. Zave, *Specifying Software Requirements.* Proceedings of the IEEE, vol. 68 ed., 1980.

[2] E. Yourdon, *Modern Structure Analysis.* Yourdon Press, 1st ed., 1989.

[3] S. Vollmann, *Aufwandschaetzung im Software Engineering.* IWT, 1st ed., 1990.

[4] M. Fowler, "Use cases and abuse cases," *www.softwarefuture.com*, June 1998.

[5] E. Kenworthy, "Use case modelling - capturing user requirements," *www.zoo.co.uk*, December 1997.