PS Software Engineering I
WS 2001/02

Project: "Info Warrior"

Peter Gruber
(Team 4)

Contents

Chapter 1

Introduction

1.1 Problem Statement

A news agency (e-news) wants to conduct collection of informations and presentation of informations with support of electronic devices. This processes should be intended to work in a quite simple way.
Potential user should mainly be seen as authors of information and users who request informations. Those requests can be made either by persons or by external systems.
This will likley result in two scenarios

- Informations are passed on to other systems by InfoWarrior. In this case, XML should be used.
- Informations are passed on to single persons. Of course, this demands an applicable presentation of informations.

An important aspect of InfoWarrior should be the easy maintenance of the system. Especially, there should be no local installation of system components.
In addition, e-news should have the possibility to prepare informations following the needs of different users; this should mean not too much work.
As a first step, InfoWarrior may be reduced to its core parts. However, additional features concerning future releasements ought to be considered.

1.2 Basic Considerations

The amount of information available today might be as huge as never before in human history. Informations are presented in various - traditional and new - ways. On the other hand, it has become more and more difficult to extract information that are relevant for individuals. For this purpose, people may refer to "preselected" forms of informations, that is newspapers or news-broadcasts on TV and radio. Nowadays, however, many people have rather immediate access to the whole "heap" of information through their modems (internet). This creates an request for WEB-based services that help keeping track of the constant flow of informations.
It is evident that this is no trivial task that is likely to lead to big and complicate algorithms. Users of such systems are likely, however, to request primarily easy accessability and fast

performance.
"Info-Warrior" may be supposed to work somewhere in this field of conflict. This might provide a background for future considerations.

Chapter 2

CRC-Session

2.1 Preparing a First CRC-Session

In this section which should be seen as an introduction to the project, we will summarize ideas and concepts we developed when thinking about how to prepare a first meeting with a client. The aim of this section is to get a rough impression what this system could look like, and to find out which questions need to be answered before going on with this project.

2.1.1 Which aims are to be pursued?

- develop a first cut modell of the project
- clarify possible misunderstandings
- articulate questions that seem important at this stage

2.1.2 Possible Candidate Classes that appear at first glance

- A receiving unit that gets informations and prepares them for storage
- A storing unit that administrates stored informations, integrates new informations and offers access to stored informations
- A request unit that handles requests to the system to look for specified informations

It may be advisible to distinguish two ways of how requests are posed to the system:

- by (personal) users that operate via a front-end system
- by other systems that make requests via electronic standards

2.1.3 Scenarios that may give an impression on how the system could work

- News-messages are transmitted to the system and now have to be prepared for storage. This has to be done very fast, so that the new informations are immediately available to users of the system.

- (Personal) users are looking for informations concerning a specific topic. The systems goes through the sored informations, selecting those which may be of interest for the user. Found informations are presented in an appealing and easy-to-overlook way, maybe according to the user's individual preferences.

- Requests for information are sent via electronic standards. Informations are selected like they are in the scenario above, but are prepared for neatly transmitting them via electronic devices.

2.1.4 Questions that still need to be answered:

- Are there any characteristics that may classify informations of different contents and styles?

- What are the main sources the system will use for receiving new informations?

- How are informations added to the storage? Who decides about articles being released or not?

- Which are typical scenarios that describe adequately the way the system will likely work?

- Who will have access to the system? On which principles should access control be based?

- Are there any aspects we have not at all considered so far?

2.2 Results of a First Meeting with Client

As a result of this first meeting, many questions were answered and a more detailed description of the system was achieved. I will summarize the most important points. This section should give a deeper understanding of the problem domain and provide a more detailed description of demands and profiles.
Data acquisition should be easy and compact, as should its presentation and input. Usually, data will be input through "manual typing", but a possibility for automatic input may also be considered.
Data acquisition is mainly done by a group of editors. Each editor oversees the work of about 10 reporters. Editors have the responsibility of controlling the reporters' work. Articles ($\rightarrow$ informations) may consist of pure text, but can also contain pictures or be designed as multimedia-presentations.
The team consists of 5 editors, each of them presiding over group of about 10 reporters. Reporters send informations every day, usually up to 5 articles every 24 hours.
Each reporter sends his articles to one editor. Editors, however, may have access to any of the reporters' articles; releasing of articles for publication is the responsibility of only one editor.
Additional characteristics that are needed to classify informations (meta-data) will be supplied. The classification of data is done through editors.

Modifications of data should be possible, as editors are responsible for controlling reporters' input and releasing them for publication. Once articles are put to the archives, their appearance should be stable.
It should be expected that the amount of stored data becomes very huge very fast. Therefore, the following modell should be considered: informations should be quickly available for one month (as long as they are up to date). When they are put to the archives, access may take a bit more time.
Access to the system should be partly limited. Informations of general interest should be freely available, but content of high importance and/or high relevance may be charged for.
Other systems that may communicate with "Info-Warrior" could include clients as the CNN-Homepage, other new's agencies, Handies (SMS) and so on. Therefore, we can assume that all those systems do already exist. The designing of interfaces needn't follow any specific instructions.
Transmission of data may be subdivided into two scenarios:

- *Pull-Method:* Users request actively informations, that is by logging in on the system.
- *Push-Method:* The systems actively sends informations to users that have subscribed for those services (e.g. e-mail, SMS, ...).

All those applications should be WEB-based, of course.
For this first stage of development, the essential tasks will be modelling the kernel of the system, more detailed aspects like accounts for users needn't be intensively discussed so far. However, the desiging of an applicable Customer-Relationship-Modell (CRM) should already be considered.
Another important aspect that was mentioned was an "Essential User-Interface Modelling". As informations needed for login (name, password, ...) should be considered, the administration of users and their authorizations should be neatly solved.
For a CRC-session next time, the following subjects should be remembered: reporter; editor; informations; user: name, address, ... ; communication-interfaces; "persons" (for purposes of inherence); ...
As a next step, Use-Cases should be discussed as potential examples of applications that may reveal the functionality of the system. Consequently, scenarios will be developed.
In a second meeting with the client, several more details of system requirements and administrative business were discussed and clarified:

- *Payment-Methods:*

  Invoice for payment concerning services that do not come for free will be done on a monthly basis. This should give a possibility to check whether users are still paying or willing to pay in time. Usually, invoices will be sent to users' postal adresses.

- The system should be developed as platform independent. The main aspect should be an easy maintenance of the system.

- *Actors:*

  The following items were agreed upon:

  - *human actors:* editors, reporters, users of the system (clients)

  - *external systems*
  - *archives:* containing news that reach further back in time
  - *storage units:* subdivided into a message storage (containing high priority information) and user data (adaministrating access control etc.)

- *entry page - user interface:*

  The entry page should present at least some accurate and on time information from all "over the world". Informations should refer to certain categories. Access to informations will be granted for different levels of payment. The user interface may also present an option to subscribe for services.

- *Sign-in-Procedures:*

  Sign-in is required for push-methods. Therefore, it will be necessary to extract informations from articles for different purposes. This should enable e.g. an SMS-News-Service, that is a minimum information in a given format.

- *Technical Requirements:*

  Editing of articles is done only once for different services. The input form of articles will thus provide seperate areas for entries such as headline, abstract (up to ... characters), categorization, payment-level of information, ... As a last instance, the editor will supervise all entries.
  Editors may compose news, too. Anyway, the two-level-process of releasement has to be secured, so any time an editor acts as "reporter", his article has to be approved by another editor.

- *Technical Aspects of Administration:*

  New subscriptions to services should be processed automatically. Thus, the system has to meet any requirements necessary. However, anytime an action concerning users of the system is taken, the administrator or people of the administration office should be informed personally. Frequent actions may be summarized as reports. Persons, however, should not be required to take actions in order to complete a standardized process.

In this second session, a first concept of User-Interface was presented and approved in general.

2.3 Results of CRC-Session

2.3.1 General Aspects

CRC (which stands for "class-responsibility-collaboration") sessions are considered a good way to extract classes from a problem statement and find out about their responsibilities and collaborations. Or, less technically: what they should do and how they will interact with other classes.
First of all, one would look for the (important) nouns in the prolem statement. Usually, they can be considered to be likely to point to a class that might be given this or a similar

name. Each card is assigned to card, that carries its name on top. Subsequently, responsibilities and collaborations of the class will be added. Evident items can be written down immediatly, but generally, those entries will be found through the execution of scenarios. Playing through those scenarios, it should become clear whether the existing model of the system - represented via the CRC-cards - is able to handle those requests, or which responsibilities and collaborations (or additonal classes) need to be supplied.

2.3.2 Scenarios during CRC-session

We will summarize the most important and significant scenarios that were discussed during the CRC-session.

- A reporter of the staff team writes a new article and submits the input to the editor for releasing procedures. Therefore, he needs an user-interface in order to communicate with the system. His article will then be stored in the Message-Storage, and the editor will be informed about the new entry (as the editor has to decide whether to release an article or not).
  As soon as the editor has decided to release the article, it can be accessed by other users who have the right to read articles. (Thus, the article needn't be stored or moved again; only its permission has to be changed (so that not only the editor, but a bigger group of persons have this permissions from that point on).)

- An user wants to look for informations about specific topics using the Info-Warrior-system. Again, a User-Interface has to provide an option for searching articles. After the user has entered his search options (categories, keywords etc.), the Message-Storage executes his request. Results will then be displayed to the user via the User-Interface. Of course, an user can only read articles that require no higher pay-level than the user is subcribed to.

- A subscribed user to the Info-Warrior-System wants to log in, so that he can use additional rights he is entitled to by subscription. After entering the necessary details via the User-Interface, the User-Storage compares his input with the system-immanent informations (especially name and password). If the log in is correct, the user now has the possibility to act as a subcribed user to the system.
  Note: This scenario may quite easily be modified for first-time log in; in this case, additonal information will be required, and the user has to be stored in the User-Storage. After first-time log in, users will have the possibility to subscribe to (billable) Info-Warrior services.

2.3.3 Results of CRC-session

As a main result of the CRC-session, a first concept of how to organize the system was found. This means, we had the possibility to some likely classes, as well as their responsibilities and interactions with other classes (collaborations).
Note: "Article" and "Message" will be used somehow synonymously from now on. "Message", however, is considered to be more general; that is, messages may consist of photos or videos, too, and needn't be restricted to written text.

- User

  responsibility: read articles, request articles,
  reporters: compose articles,
  editors: edit artciles, release articles

  collaborators: User Interface

- User-Interface

  responsibility: provide operations to users: log in, read articles, search articles, ...

  collaborators: User, Message Storage, User Storage

- Message Storage

  responsibility: store new messages, look for stored articles, provide access to stored articles

  collaborators: User-Interface

- User Storage

  responsibility: store informations about users, manage log in procedures, provide informations about users' rights and subscriptions

  collaborators: User-Interface, Message-Storage

In order to manage communication among the different units, we considered implementing two abstract classes:

- *Request:* a standardized format for all kinds of actions users may want to perform on the system (read articles, compose articles, ...).
- *Message:* a standardized format for all kinds of informations that might be stored in the system. This should give a possibility to classify stored articles along certain criterias, even though there are likely differences in lay out, content etc. A message will have criteria, too, that make it possible to connect articles to authors, date etc. and manage to process of publication (messages can be marked "released" or "submitted" (if reporters submit their articles to the editor) for example.

Chapter 3

User-Interface-Modelling

A very important aspect for InfoWarrior is the User-Interface, as the system will substantially work as front-end-system. In a meeting with the client, we presented our suggestions which will be summarized in the next paragraphs.

3.1 Description of the User Interface

In this part, the most important web-pages and its features will be presented.

- *Start Page:* It will present the most recent news which will be of interest for most "typical" users. In addition, it will provide a log in-field for registered users and the option to create a new account (first-time log in, sign up). And, of course, it will offer an link to search for informations.
- *Sign-up Page:* On first-time log in, users are asked to enter necessary, individual data (name, password, ...).
- *Search Page:* Users may select several options to specify what they are looking for. For example, search for informations can be restricted to certain categories, articles that were released during a special period of time, articles containing certain keywords etc. Search-Results will be displayed on a seperate page.
- *Search-Results Page:* Articles matching defined criteria will be presented. Of course, the user has the possibility to access these articles directly. If necessary, he may conduct a secondary search that is restricted to the results already presented.
- *Article Page:* It displays stored articles in an appaeling look and feel. As an additional feature, links to related articles are provided.
- *Preferences Page:* Users who are registered in the system can set individual preferences (e.g. lay out of displayed articles). This page will provide corresponding options.

For reporters and/or editors, some more important page have to be considered:

- *Compose Articles Page:* This page should provide some functionality for reporters who write articles. It should give each reporter a summary on the articles he is currently working on, as well as other important informations (e-mails, for example). It will also provide an option to submit articles to the editor when finished.

- *Edit Articles Page:* Editors should be given a summary on articles that were written by their assigned reporters. Look and feel will be somehow similar to the Compose Articles Page, editors, however, will also have the option to release articles for publication.

In group work, HTML- and WML-based prototypes for User-Interfaces were developed, too. We tried to observe the following design principles:

- All pages should have a consistent look and feel.
- Each page should feature a navigation element, so that users have a good overall view. One aspect is a constant title bar for all pages.
- Important functions should be easily accessible.

Screen-shots of these two prototypes can be found in the group-document.
During presentation and discussion of the HTML-based version, some more demands to the system emerged:

- *Search-Link:* A search can be restricted to categories. In addition to the basic search page, more sophisticated pages may be developed, too. Repeated search might be optimized by restricting the search to previous search results.
- *Subscriptions:* Payment levels can be selected seperately for each category. Therefore, we will have to discuss one more usecase: reconfiguring information about users.
- *Articles:* A reporter has the possibility to clear, save and submit articles (email them to editor). The editor will finally decide about payment level and categorization.
  Articles that are submitted to an editor should be linked to the names of reporters rather than their identification numbers.
  As related articles should be linked together, additional meta-information on articles is required.

Discussing the WML-based version, the following important aspects were mentioned:
Different layouts of these two prototypes demand different flow charts. Thus, one has to look for elegant ways to handle different technologies. As the logic background is the same, however, it might be enough only to change the view and keep the rest the same.

Chapter 4

Use Cases

In this chapter, use cases that cover the most important aspects of the Info-Warrior system will be discussed.

4.0.1 Description

1. **Name:** Sign-up to InfoWarrior Services (first-time log in)
   Description: An user wants to sign-up to Info-Warrior services. He clicks the sign-up button on the start page. Then, he will be presented a sign-up page where he has to enter all necessary information, that is mainly: name, address, e-mail-address, password and so on. After that, the user is assigned an account. He can subscribe to extended Info-Warrior services.

2. **Name:** Subscribe to Info-Warrior Services
   Description: An user wants to subscribe to Info-Warrior services. He clicks the subscribe button on the start page. On the subscription page, the user can select several options for subscriptions (categories, pay-level, duration etc.). After that, user-data in the user-storage has to be modified. The user can now access articles that require the subscription he has just chosen.
   Note: In a quite similar ways, users may unsubscribe from Info-Warrior services, too.

3. **Name:** User logs in
   Description: An user opens the Info-Warrior start page and enters name and password. This data are checked by the User Storage for correctness. If log in is successful, the User-Interface that is presented may be modified depending on the user's role. If the user is assigned as reporter, for example, the page will offer specific options (e.g. compose articles). Anyway, each user who is logged in will have the possibility to set his individual preferences, for example.

4. **Name:** Reporter composes article
   Description: First of all, the reporter has to log in on the system as described above. After that, he can choose the option compose article. Consequently, the message storage is searched for articles that were written by this reporter and were not yet submitted to the editor. This list is presented to the reporter, who can now choose whether to resume work on one of the found articles or start writing a new one. When he has finished, he can store the article in the system or submit his

articles to the editor. (This means, this article is internally marked "submitted" by the system.)

5. **Name:** Editor releases article
   Description: In the same way as above, the editor is presented a list of articles that were submitted for publication. He can choose one of those articles in order to read or edit it. Finally, he can decide to release this article for publication. (Again, this means the article is marked "released" by the system.)

6. **Name:** Search informations
   Description: An user wants to search informations. He clicks the search-button and enters the search criteria or selects options on the search page. Now, the Message Storage is Searched for corresponding informations. The results are presented via the Search Results Page. Users who want to search informations needn't necessarily be logged in, but users who have subscriptions can access this greater scope of informations only when logged in, of course.

7. **Name:** System Administrator's Task
   Description: In order to maintain the system, it will be necessary to appoint an administrator who can conduct more sophisticated operations. At this point of analysis, however, it seems neither to be useful nor possible to define his tasks in detail.

Chapter 5

Class Diagrams

Based on the results of CRC-session and use-cases discussion, a Class Diagram was developed in team work which was then included in the group document. Here, a description of the (relevant) classes, their operations and attributes and the communication between them will be given.

- **User:** Thus class represents physical users in the system. Users are described mainly by the following attributes: Hame, Address, EMail, Phone, CreditCard, ... As users may log in on the system, log in informations like password also have to be stored. If users have a subscription to Info-Warrior services, this class has to interact with the class "subscriptions" (user has a subscription).

- **Subscription:** A Subscription is characterized by its attributes PayLevel and Category. Each subscription object is assigned to a user.

- **Storage:** This is an abstract class which provides basic features for both, User-Storage and MessageStorage.

- **User Stroage:** User data is stored in this class. This class has to provide search operations if user data is needed and needs to interact primarily with the MessageCore in order to check if users have the right to access articles or not.

- **Message Storage:** This class manages the storage of articles that are handled by the system. It has to provide operations for adding and deleting informations and, of course, has to implement the search operation if users want to look for information that is stored in the Info-Warrior System.
  The Message storage is subdivided into a PrimaryMessageStorage (most recent messages that need to be easily and fast accessible) and a SecondaryMessageStorage (archives).

- **Message:** Message represents any kind of information, especially articles, that are to be stored in the Info-Warrior System. It provides a uniform scheme for stored informations (Title, Abstract, Text, ...), including attributes that are necessary for access control (PayLevel) and the process of publication (articles are stored, submitted (to the editor) or finally released for publication).

- **MessageCore:** This class handles any requests that require informations or actions from the MessageStorage (e.g. search operations). Thus, it has to interact with the abstract classes Storage and Request which provides a standardized form how requests to the system are encoded. MessageCore also has to return results of operations, of course.

- **UserCore:** This class handles any requests that require informations or actions from the UserStorage Unit. It works similarily to the MessageCore.

- **Request:** Any requests to the system should have a standardized form which is implmented by this class. Thus, it has to interact with the SystemInterface as well as UserCore and MessageCore.

- **SystemInterface:** The SystemInterface has to receive requests that are posed tothe system from "outside" and return an appropriate response to the client. As those requests will be posed via XML, SystemInterface uses an XML-converter to translate those requests to the uniformed Request-scheme.

- **ExternalSystem:** ExternalSystem is an abstract class that represents any kind of outside systems that interact with Info-Warrior.

- **UserInterface:** The UserInterface is one special, but probably the most important ExternalSystem. It offers (physical) persons several ways to interact with the Info-Warrior system. It has to be adapted to the needs of different users (customers, reporters etc. - compare the UserInterfacePrototyping). The UserInterface must directly communicate with the SystemInterface.

- **ACLSecuredObject:** Most operations offered and informations stored in the Info-Warrior system are available to a limited group of users. Access is controlled via AccessControlLists. Whenever such requests are posed to the system, it has to be secured that the user (anonymous or logged in) is entitled to conduct these operations. In order to check those rights, this class needs to interact with Message- and UserCore (which execute requests) as well as with User and Message (which provide relevant informations).

Chapter 6

Sequence and Activity Diagrams

Based on the results of the CRC session, the description of use cases and the class diagram, sequence diagrams and activity diagrams were composed in team work. Their main task is to graphically present the information that is given by the description of use cases and the class diagram. That is, to demonstrate the actual flow of data between classes.
Sequence Diagrams, as well as Class Diagrams, are included in the group document.