Петрозаводский государственный университет
Математический факультет

Кафедра информатики и математического
обеспечения

Курсовая работа

Система Web-SynDic: Разработка программного обеспечения на основе прецедентов

Выполнил: студент 3 курса
Ананьин А.В.

Научные руководители:
к.т.н., доцент Богоявленский Ю.А.
к.ф.-м.н., ст. преподаватель Корзун Д.Ж.

Представлена ___ _____ 2004 г.

Оценка промежуточного отчета:

Оценка научного руководителя:

Оформление и срок
представления:

Итоговая оценка:

Отв. преподаватель:

Петрозаводск
2004

Содержание

1 Введение

Неотрицательными линейными диофантовыми уравнениями (НЛДУ) называются линейные уравнения с целыми коэффициентами и с решениями в неотрицательных целых. Они являются актуальным объектом научных исследований в теории чисел, теории полугрупп и теории алгоритмов, а также

находят важные приложения в задачах целочисленного программирования, исследования операций и кибернетики.

Целью проекта Web-SynDic [1] является разработка web-системы для демонстрации и тестирования синтаксических алгоритмов решения систем АНЛДУ (ассоциированная с КС-грамматикой система неотрицательных линейных диофантовых уравнений) [2]. Web-система позволяет исследователям через Интернет задавать вручную или генерировать автоматически системы АНЛДУ, находить их базис Гильберта, проверять правильность результата, оценивать потребление ресурсов и сравнивать эффективность алгоритмов.

Одной из важнейших задач проекта была задача разработки модели прецедентов (на стадиях анализа требований и проектирования), а также программной реализации и тестирования системы на основе этой модели [4].

Под прецедентом понимается логически завершенная последовательность событий, связанных с исполнителем (внешним объектом), который для выполнения требуемой функции использует программную систему. Примером прецедента в системе Web-SynDic может быть прецедент "Solve ANLDE System" ("Решить систему АНЛДУ"), а внешним объектом является Solver (программа, реализующая алгоритм решения АНЛДУ-систем). На основе языка UML модель прецедентов включает в себя диаграмму прецедентов и описание каждого прецедента в отдельности с соответствующими диаграммами последовательностей [5] [6].

В дальнейшем, в ходе описания последующих разделов будет использоваться набор документации системы Web-SynDic на английском языке.

2 Построение модели прецедентов

В рамках проекта Web-SynDic построение модели прецедентов (use case model) заключалось в следующем: На стадии анализа требований составлялся список высокоуровневых требований (User Requirements) заказчика.

2.1 User Requirements: Version 1.20

The aim of the Project is development of a web system for demonstrating the work of the syntactic algorithms for solving ANLDE systems. The users of the web system are researches that have an access to the Internet and use standard Internet browsers. The main objectives of the web system are the following.

- Presenting the key facts of the ANLDE systems theory for the international scientific community.
- Demonstrating and testing the efficiency of the syntactic algorithms with the visual tool on various examples including user's ones.

- Providing the student team with the experience what the process of competent development of real software systems is.

2.1.1 Development process

G1. The web system must be developed according with international principles, practice, standards and recommendations of SE. Use Pressman, Sommerville, Larman, Conallen as basic books.

G2. The Project starts on 16.07.2003 and has to be completed in November, 2003.

G3. Working language of the Project is English. All artifacts are replicated in Russian.

G4. From the start of the project, students have to work 20 hours/week.

2.1.2 Functions of the web system

F1a. The web system must solve an input ANLDE system (homogeneous) and show to a user the outcome. The input ANLDE system is given by a user or generated by the software (user choice). These input systems are considered as test ANLDE systems. Only homogeneous ANLDE are used.

F1b. For the case of Requirement F1a the outcome must include:

1. Test ANLDE system.
2. Solutions of the ANLDE system, e.g. Hilbert basis or a particular solution.
3. Metrics of the resource consumption by the syntactic algorithm (time and memory usage estimates);
4. Comparative metrics for the alternative algorithms (slopes, lp_solver, BonsaiG, GLPK).
5. Key hardware characteristics of the server.

F2a. The web system must solve a set of test ANLDE systems (homogeneous). The input test set is given by a user (TXT file) or generated by the software (user choice). Only homogeneous ANLDE are used.

F2b. For the case of Requirement F2a the outcome must include:

1. Characteristics of the input set of ANLDE systems.
2. Statistics of the resource consumption by the syntactic algorithm (time and memory usage estimates).

3. Comparative statistics for the alternative algorithms (see references for these algorithms in Requirement F1b).
4. Key hardware characteristics of the server.

F3. The web system must allow a user to send her/his opinion on the solution result. A special case here is user's explicit disagreement with the found solution(s) of the processed test ANLDE system (testing of the syntactic algorithms). The test ANLDE system may be included to the opinion.

F4. The web system must register a user when she/he wishes. A registered user has a unique identifier (nick name).

F5. The web system must compute activity statistics of the registered users including time and resource consumption (available for system administrator only).

F6. The web system must compute activity statistics of all users. These users are identified by their IP-addresses (available for system administrator only).

2.1.3 Attributes of the web system

Usability

AU1. The traditional mathematical style must be supported for representation of ANLDE (and possibly NLDE) systems and their solutions.

AU2. The output must be available to a user in HTML and TXT formats.

AU3. Standard Internet browsers (among them Netscape, Mozilla, MS-IExplorer) must be supported.

Security

AS1. The demonstrated algorithms (ANLDE system solvers and test generators) must not be accessible to the external side. Only the outcomes of their work are available.

AS2. There are two types of users: regular ones and a system administrator.

AS3. The activity statistics must not be accessible for regular users.

AS4. For any regular user the web system must support default limits on the solution process (maximum time, memory, absolute values of coefficients, maximum number of equations, unknowns, ANLDE systems in a test set, solutions in Hilbert basis, etc.).

AS5. A regular user may manage her/his own limits on the solution process; these limits must not exceed the default ones (see Req. AS4).

Performance

AP1. The web system must serve concurrently up to 5 users (separate user sessions) without significant reduction of the performance.

AP2. The web system must not overload a base web server more than 75% of the total server workload.

AP3. The web system must reply on a user action less than after 20 seconds. The reply is either the required data, or a notification on the progress.

Deployment

AD1. Client part of the web system must be available to a user via an Internet browser without an explicit installation.

Затем на основе этих требований создавались расширенные (expanded user requirements) и системные (system requirements) требования.

Пример расширенного требования:

2.2 EU1a: Solving a test ANLDE system

Description

The web system solves a test ANLDE system and produces the report on solution.

User Actions

1. Start the ANLDE processing subsystem, select solver (AU3, AP1, AP2, AP3).
2. Input a test ANLDE system (defined by a user or generated automatically) (AU1, AU3, AS1).
3. Send the test ANLDE system to the server (AU3, AP1, AP3).
4. Wait for the report on solution (AS4, AS5, AU3, AS1, AP1, AP2, AP3).
5. View the report on solution (F1b, AU1, AU2, AU3, AS1, AP1, AP2).

References

System requirements: inputANLDESystem; generateANLDESystem; sendANLDESystem; solveANLDESystem; sendProcessMessage; sendANLDESystemReport; loadANLDESystems.
Use cases: Process a test ANLDE system.

Пример системного требования:

2.3 sendAcknowledgments

Description: This function sends different types of acknowledgments.

Author: Andrew V. Ananin
http://zeta.cs.karelia.ru/Web-SynDic/doc/eng/personal/ananin.html

Input Data: Information for outputting in acknowledgment message.

Sources of Input Data: Function sendANLDESystem, Function sendANLDESystemSet, Function registerUser, Function sendUserNotes, Function manageUserLimits, Function manageDefaultLimits, Function manageUsers.

Output Data: Acknowledgment message.

Destinations of Output Data: Client part (browser).

Function requires Client part (browser).

Preconditions: Function sendANLDESystem] must be executed, Function sendANLDESystemSet must be executed, Function registerUser must be executed, Function sendUserNotes must be executed, Function manageUserLimits must be executed, Function manageDefaultLimits must be executed, Function manageUsers must be executed.

Postconditions: None.

Restrictions: Req. F1a, F2a, F3, F4, AU3, AP3, AD1.

Side Effects: None.

Moot Points: None.

Risks: Functions sendANLDESystemSet, manageUserLimits, sendAcknowledgments may be not implemented.

Priority: Secondary.

Каждое расширенное требование определяло отдельный прецедент. Затем выполнялось построение диаграммы прецедентов на основе выделения внешних объектов и взаимосвязей между прецедентами, а также строились описания каждого прецедента в отдельности с соответствующими диаграммами последовательностей.

2.4 Use cases

Use case model is the one of the most important views on the required web system functionality because it combines the expanded user requirements and system requirements. Therefore, this is the most comprehensive model for the functions of the developing web system.

There are two main types of actors: a user and an external algorithm. The other actors are their descendants: regular user, registered user, and system administrator (users); solver and generator (external algorithms).

The identification of the use cases is mainly based on the expanded user requirements. Each expanded user requirement corresponds to one use case.

Analysis of each use case is based on the system requirements. They form all high-level operations of the web system (see the corresponding sequence diagrams).

Рис. 1: High-level use cases diagram

Пример описания прецедента

2.5 Manage users

2.5.1 Author

Andrey V. Anan'in

Textual description

User management starts when user logs in as sysadmin and chooses management on the web page. The web system sends form for input a user's nickname. Then he inputs a nickname and sends the request on search. The web system searches this user in data store. If there are not corresponding user record in the data store, the web system sends report on absence this user. Otherwise the web system sends form with user's information. Sysadmin can change this information or remove the user profile. Then the form is sent to the web system, and it performs requested operation and sends an acknowledgment to sysadmin.

2.5.2 High-level description

Use case	Manage users.
Actors	Sysadmin.
Description	Sysadmin can change or delete information about registered users.
References	User requirements: F4, F5, F6, AS2. Expanded user requirements: EU2a, EU3b, EU3a. System requirements: manageUsers (primary), sendAcknowledgments (secondary).

2.5.3 Sequence diagram

See Figure 2.

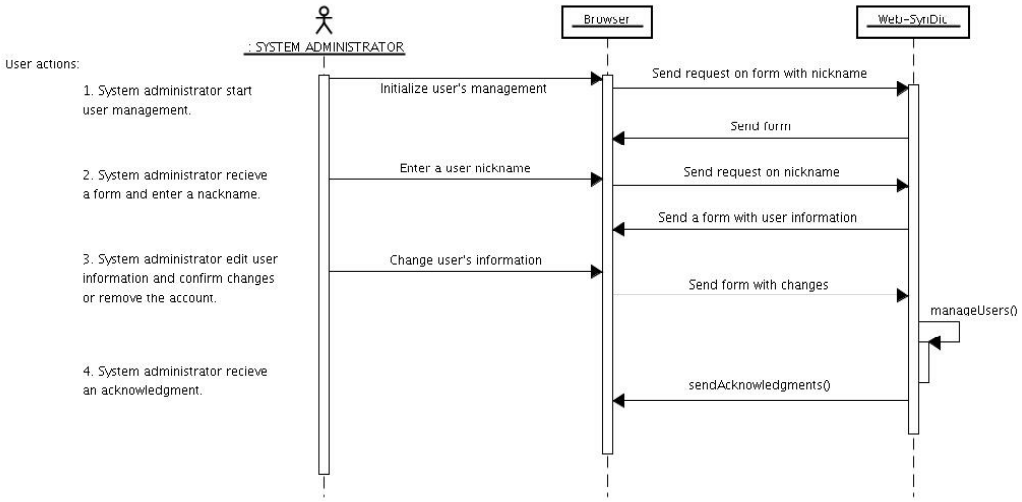

Рис. 2: Sequence diagram for use case **Manage users**

Построенная таким образом модель прецедентов в дальнейшем активно использовалась на других стадиях разработки.

3 Использование модели прецедентов на различных стадиях разработки

На стадии проектирования на основе построенной модели прецедентов были созданы модель архитектуры системы, поведенческая модель, а также формы, которые являются основной частью пользовательского интерфейса системы Web-SynDic [3].

Для создания модели архитектуры была проанализирована модель прецедентов и в результате были спроектированы и реализованы подсистемы, которые непосредственно реализовывали каждый прецедент.

3.1 Data store

The class diagram used in data store is shown in Fig. 3.

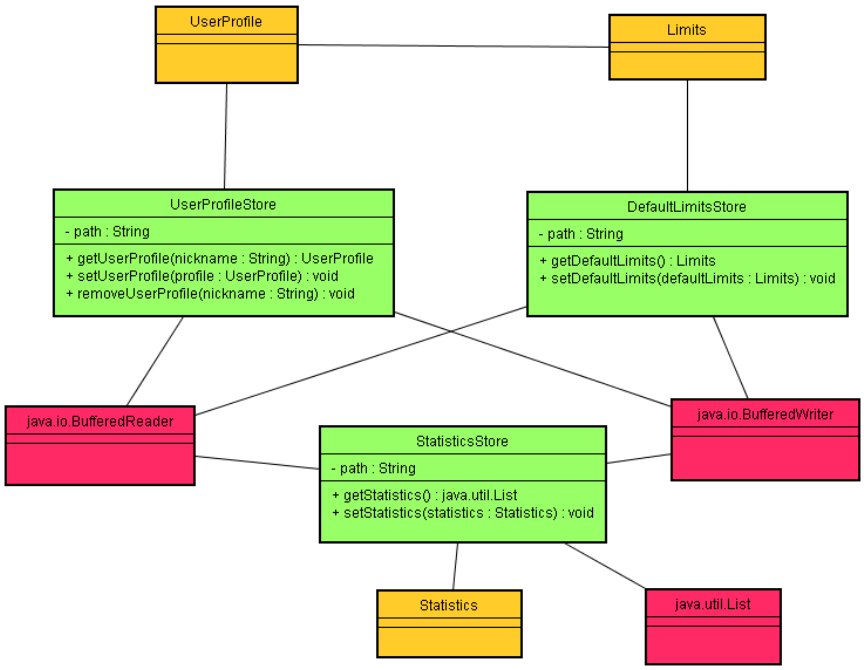

Рис. 3: Data store architecture

The data store has three interface classes for user information, default limits information and statistics information from the data store. It realizes writing and reading data from the data store from the hard disk. Information about users, default limits and statistics is storing in log files (Statistics log file, default limits log file, user profile log files). First string in each files is reserved for a format version number.

3.1.1 User profile log file

Description: This section describes user profile file format. The user nickname and the name of corresponding user profile file are the same.

Format: A user profile log file is a set of strings. First string after format version number is a user's password. Next string is email. The next nine strings are the user limits (Limits format). And all remaining strings are information about a user.

Example:

#version: 1
qwerty
qwerty@localhost.localdomain
100
3000
100
100
20
20
20
1000
50
I live in Petrozavodsk

3.1.2 Class UserProfileStore

Description: Class for writing, reading and/or deleting user profiles.

Fields:

private String path; — path to the user profiles store.

Constructors :

public void UserProfileStore(String path); — initializes object with path to the user profile store.

Methods:

public UserProfile getUserProfile(String nickname);— gets user profile from the hard disk with corresponding nickname
public void setUserProfile(UserProfile profile); — writes user profile on the hard disk
public removeUserProfile(String nickname); — deletes user profile

3.1.3 Class DefaultLimitsStore

Description: Class for writing and reading default limits.

Fields:

private String path; — path to the default limits file

Constructors :

public void DefaultLimitsStore(String path); — initializes object with path to the default limits file

Methods:

public Limits getDefaultLimits(); — gets default limits from the data store
public void setDefaultLimits(Limits defaultLimits); — writes default limits on the data store

3.1.4 Default limits file

Description: This section describes default limits file format.

Format: A default limits file consists of a set of strings. Each string is a one default limit (Limits format).

Example:

#version: 1
100
3000
100
100
20
20
20
1000
50

3.1.5 Statistics log file

Description: This section describes statistics file format. Each file contents monthly activity statistics.

Format: A statistics log file is a set of strings. Each string is a statistics for one user. String has a set of fields separated by white spaces. Format described in Statistics class. Fields in the string are:

1. user nickname
2. user IP address
3. number of generated systems
4. number of input systems
5. number of solved systems
6. number of acknowledgment systems
7. number of systems with solutions, which are discrepances solving
8. summary system time(sec)
9. summary user work time(sec)
10. summary used memory(Kb)
11. start session time mark(ms)
12. end session time mark(ms)

Example:

#version: 1
guest 123.123.123.123 15 13 14 9 1 0.1 9.34 2192 1068376393 1068377093

3.1.6 Class StatisticsStore

Description: Class for writing and reading activity statistics. Statistics is collected in a buffer. Records of statistics are writed on the data store per month. Each file of statistics is a monthly statistics. Activity statistics is accessible via web-interface only for a current month. Statistics for other previous months is stored on the hard disk of the server.

Fields:

private String path; — path to the statistics files

Constructors:

public void statisticsStore(path); — initialize object with path to the statistics files

Methods:

public java.util.List getStatistics(); — get statistics from the data store, returns list of Statistics objects
public void setStatistics(Statistics statistics); — write monthly statistics to the buffer and data store from the buffer.

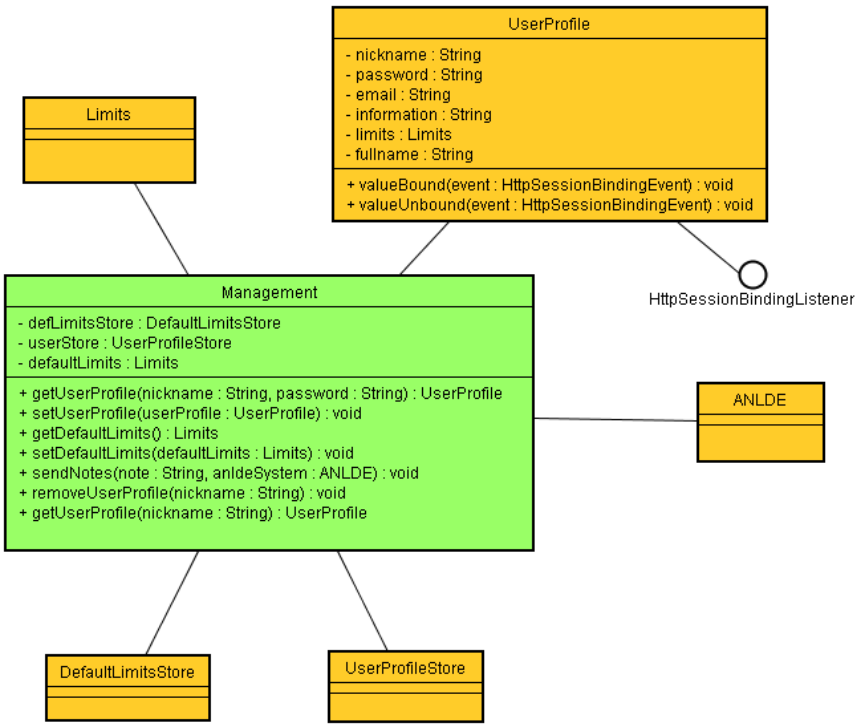

Рис. 4: Management architecture

3.2 Management

The class diagram used in management subsystem is shown in Fig. 4.

Management has one interface class. It realizes different management functions.

3.2.1 Class UserProfile

Description: Class for storing user profile.

Fields:

private String nickname; — user's nickname
private String fullname; — user's fullname
private String password; — user's password
private String email; — user's email
private String information; — user's information
private Limits limits; — user's limits

Methods:

public String getNickname(); — gets user's nickname
public void setNickname(String nickname); — sets user's nickname
public String getFullname(); — gets user's full name
public void setFullname(String nickname); — sets user's fullname

public String getpassword(); — gets user's password
public void setpassword(String password); — sets user's passwords
public String getEmail(); — gets user's email
public void setEmail(String email); — sets user's email
public String getInformation(); — gets user's information
public void setInformation(String information); — sets user's information
public Limits getLimits(); — gets user's limits
public void setLimits(Limits limits); — sets user's limits
public void valueBound(HttpSessionBindingEvent event); — empty finction for interface realization
public void valueUnbound(HttpSessionBindingEvent event); — notifies user profile to save itself

3.2.2 Class Management

Description: Class for management. Functions in this classes manage default limits and users profiles. It is not necessary to use separate functions for managing user limits, because limits are stored in UserProfile class.

Fields:

private DeafultLimitsStore defLimitsStore; — object for default limits
private UserProfileStore userStore; — object for user profile
private Limits deafaultLimits; — default limits

Constructors :

public void UserProfileStore(); — initializes object with defLimitsStore and userStore objects

Methods:

public UserProfile getUserProfile(String nickname, String password); — checks accordance of nickname to the corresponding user profile log file; checks password; gets user profile; constrain user limits by default limits
public UserProfile getUserProfile(String nickname); — gets user profile
public void setUserProfile(UserProfile profile); — save userProfile
public delUserProfile(String nickname); — deletes user profile
public Limits getDefaultLimits(); — gets default limits
public void setDefaultLimits(Limits defaultLimits); — writes default limits
public void sendNotes(String note, ANLDE anldeSystem); — managing notes, send notes to the system administrator e-mail

Одной из главных частей системы Web-SynDic является пользовательский интерфейс, в основе которого лежат формы. Формы также создавались на рснове прецедентов, а точнее на основе системных требований.

Пример формы

3.3 Forms

3.3.1 Log In

Description: Log In form allows a registered user to identify her/himself in the web-system. A registered user enters her/his nickname and password. There is a user who has an opportunity to Log In the web-system as a system administrator with access to the administration forms. A "Register"button is also available; thus using this form, a user can register her/himself whenever she/he wants. See Fig. 5.

Рис. 5: Log In form

Components:

- text field for nickname (User Information format),
- text field for password (User Information format),
- button "Log In",
- button "Register".

References:

Use case: Log In.

Requirements: EU2d.

Также поведенческая модель системы, которая описывает различные варианты использования системы реализует прецеденты.

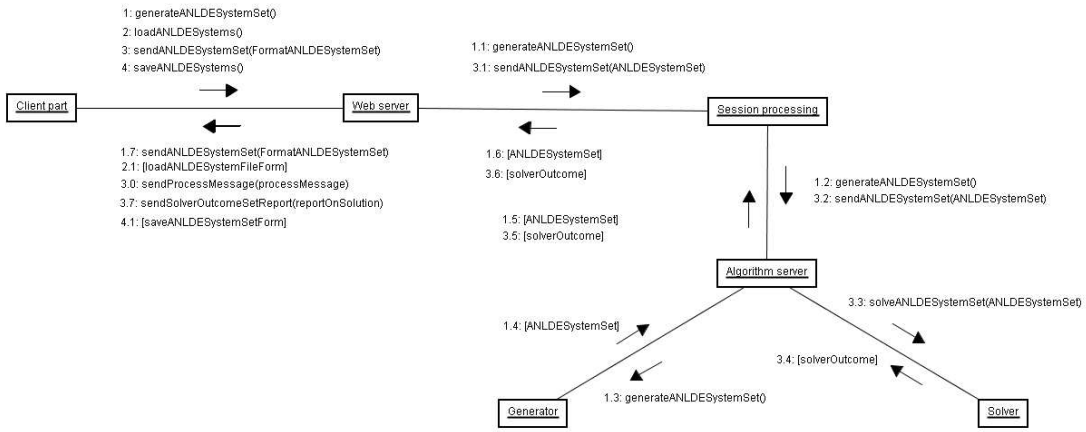

Рис. 6: Process a set of ANLDE systems Collaboration Diagram

3.4 Behavioral model

3.4.1 Process a set of ANLDE systems

For processing a set of test ANLDE systems, a user may initialize four flows, see Figure 6.

The first flow is for generating such a set.

1: generateANLDESystemSet() sends a signal to the web server that a user requires to generate a set of test ANLDE systems,

1.1, 1.2: generateANLDESystemSet() sends/forwards the request to the algorithm server,

1.3: generateANLDESystemSet() calls an appropriate generator and starts the generating process,

1.4: [ANLDESystemSet] is a required generated set of ANLDE systems,

1.5: [ANLDESystemSet] is forwarded to the session,

1.6: [ANLDESystemSet] is returned to the web server,

1.7: sendANLDESystemSet(ANLDESystemSet) sends the form with the set of ANLDE systems to the user.

The second flow is for loading a set of ANLDE systems from user's file.

2: loadANLDESystem() sends request to getting a web form for loading a set of ANLDE systems,

2.1: [inputANLDESystemForm] (the web form) is sent to the user for loading a set of ANLDE systems.

The third flow is for solving a set of ANLDE systems.

3: sendANLDESystemSet(FormatANLDESystemSet) sends a given set of ANLDE systems in ANLDE format to,

3.0: sendProcessMessage(processMessage) sends a process message (about solving process),

3.1: sendANLDESystemSet(ANLDESystemSet) sends a given set of ANLDE systems to the web server,

3.2: sendANLDESystemSet(ANLDESystemSet) forwards the set to the algorithm server,

3.3: sendANLDESystemSet(ANLDESystemSet) calls an appropriate solver and starts the solving process,

3.4: [solverOutcome] (solution result) is returned by the solver to the algorithm server,

3.5: [solverOutcome] is processed and sent into the current session,

3.6: [solverOutcome] is processed and sent to the web server,

3.7: sendSolverOutcomeSetReport(reportOnSolution) produces a report on solution and sends th corresponding form to the user.

The last flow is for saving ANLDE system.

4: saveANLDESystems() is a request for saving a given set of ANLDE systems,

4.1: [saveANLDESystemForm] (the form for saving) is produced and sent by the web server to user.

A user chooses processing a set of ANLDE systems using the corresponding item in the main menu. The "Process a set of ANLDE systems" is displayed in the "Content" part of the main web page.

The first possibility for user is to input a set of ANLDE systems. The set may be loaded from file. The user chooses "Load set from a text file and solve" item in the form. Then the user presses the "Browse..." button and chooses the file. Also the user can choose an alternative solving algorithm (corresponding item in the list "Alternative Solver").

The second possibility is to generate a set of ANLDE systems. The user chooses "Generate new set" item. Also the user can choose a generator (corresponding item in the "Generator"). If the user does not choose a generator, then "gauss" algorithm is used by default. After the generation, the user can choose solve and/or save a set of ANLDE systems by selecting corresponding check boxes.

For processing a set of ANLDE systems, a user should press the “Process” button. The process message will be displayed in the “Content” part (sect. Process message format). Then the web system checks the given set of ANLDE systems. If the set is incorrect, then an error message is displayed in “Content” (sect. Error message format, 4). If the set does not satisfy user limits, then an error message is also displayed in “Content” (sect. Error message format, 7).

If save item was selected, then after generating ANLDE systems the set is opened in a new browser window (ANLDE system set format, comments on saving will be added). If no check box is selected, then an error message is displayed in the “Content” part (sect. Error message format, 5).

If the set of ANLDE systems is valid, then the report on solution is displayed in the “Content” part after processing (sect. ANLDE system set solution format). If there was an error in the solving process, then the error is displayed in the report on solution (sect. ANLDE system set solution format).

For convenience, user limits information is shown in this form too. If the user presses “Change limits” button, then the “User limits” form is displayed in the "Content"part and, after submitting changes "Process a set of ANLDE systems"form, is loaded in the "Content"part with new user limits.

This form corresponds to the "Process a set of ANLDE systems"use case, see Fig. 7.

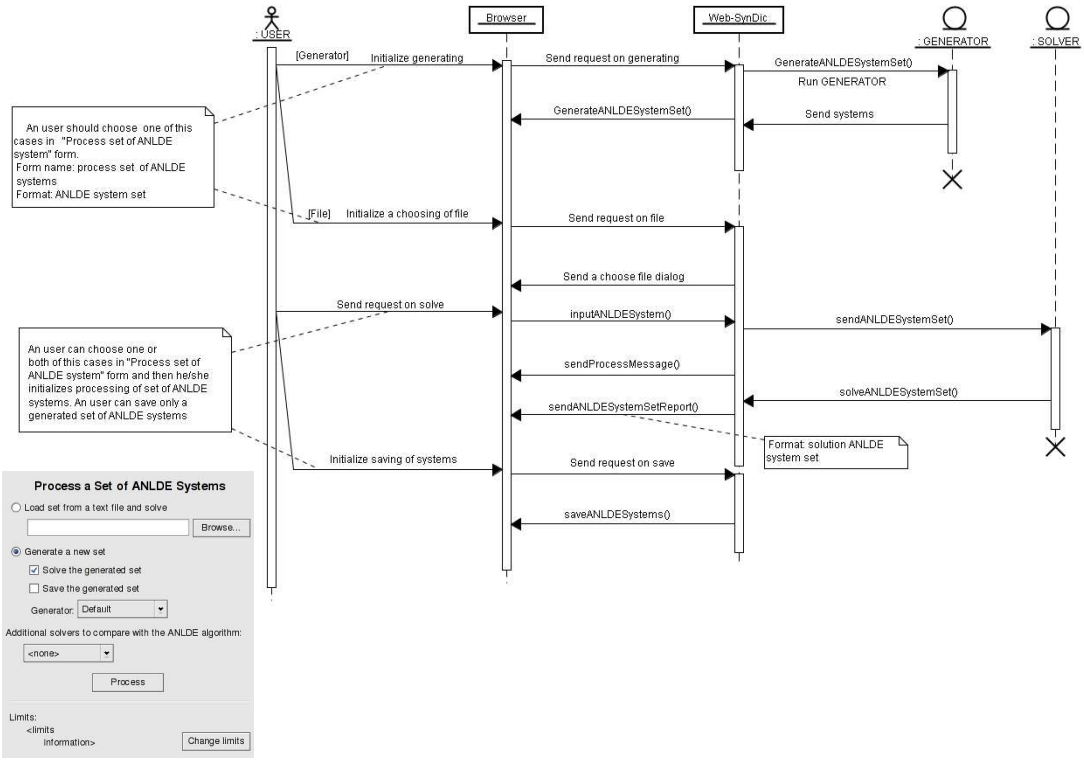

Рис. 7: Process a Set of ANLDE Systems

На стадии тестирования были разработаны базовые системные тесты, основой которых также являлись прецеденты, а именно: было проверено, как реализованы прецеденты, и какова реакция системы на различные действия пользователей.

Пример базового системного теста

3.5 Process a set of ANLDE systems

Description: The web system solves set of ANLDE systems and produces the report on solution.
Type of test: functional, acceptance.
Precondition: One of the Web-SynDic system pages is opened in the user's browser.
Testing:

1. UI process form

   User Actions: The user clicks on Process a Set of ANLDE Systems link in Main Menu.

   Expected Outcome: The user is returned with Process a Set of ANLDE Systems form in the Content area.

2. ANLDE systems set inputting

   User Actions:

   Generation: User can set Generate a new set radio button, Solve generated set or Save generated set in Process an ANLDE System form.

   Input: User can set the Load set from a text file and solve radio button and manually enters path to text file containing a set of ANLDE Systems to solve in text area, or clicks on Browse button and chooses file in the appeared browse window.

   Expected Outcome The corresponding forms and radio buttons are filled properly.

3. Solving

   User Actions:

   Standard: the user clicks on Process button of Process a Set of ANLDE System form.

   Comparing: the user sets slopes checkbox, or any other checkbox corresponding to one of external ANLDE solvers, and clicks on Process button of Process a Set of ANLDE System form.

Expected Outcome: If generation or processing of the set of ANLDE systems takes more than 20 seconds, user is returned with Progress Message every 20 seconds until processing is done.

When processing is done, the user is returned with following results, depending on set parameters.

Set	Result
Load set from a text file and solve radio button	report on solution
Generate a new set radio button and Solve generated set checkbox	report on solution
slopes checkbox, and any of two previous cases	report on solution: internal algorithms and external solvers
Generate a new set radio button and Save generated set checkbox	new browser window with the generated set of ANLDE systems

Example test data:

User Input

```
x1 + x4 = 2*x1 + 3*x3
x2 + x3 = x1 + 2*x2 + x3
%
x1 + x11            = 82*x1 + 56*x2 + 48*x5 + 22*x8 +
                      19*x10 + 49*x11 + 92*x12
x2 + x7 + x9 + x12 = 50*x1 + 48*x2 + 18*x5 + x7 + 20*x8 +
                      x9 + 86*x10 + 71*x11 + 30*x12
x3 + x10 + x14      = 15*x1 + 66*x2 + x3 + 83*x5 + 76*x8 +
                      74*x10 + 96*x11 + 62*x12 + x14
x4 + x8             = 29*x1 + 29*x2 + x4 + 28*x5 + 86*x8 +
                      36*x10 + 74*x11 + 5*x12
x5 + x6 + x13       = 51*x1 + 26*x2 + 3*x5 +x6 + 33*x8 +
                      81*x10 + 49*x11 + 54*x12 + x13
%
x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 +
x9 + x10 + x11 + x12 + x13 + x14
                    = 90*x1 + x2 + 7*x3 + 88*x4 + 96*x5 +
                      59*x6 + x7 + 7*x8 + x9 + x10 + 63*x11 +
                      55*x12 + x13 + 89*x14
%
x1       = x1
x2 + x3 = x2 + x3
```

Expected Solution

1. ANLDE system metrics:

```
minimum number of equations: 1
average number of equations: 2.5
maximum number of equations: 5

minimum number of unknowns: 3
average number of unknowns: 8.75
maximum number of unknowns: 14

maximum coefficients in ANLDE systems: 96

minimum number of solutions: 1
average number of solutions: 4
maximum number of solutions: 7
```

2. Algorithm metrics:

Algorithm	System time(sec)	Work time(sec)	Memory usage(Kb)	Solving result
anlde	0.00	1.513098	2192	Solved
slopes	0.72	3.483165	2192	Abnormal solver termination

3. Solving machine characteristics:

   - **CPU:** IA32, 1200 MHz;
   - **RAM:** 256 MB.
   - **Operating system:** Linux 2.4.19
   - **Priority(nice):** 15.

Руководство пользователя web-системы также строилось по прецедентам.

4 Заключение

Модель прецедентов является мощным средством, используемым при разработке программного обеспечения. Она не совпадает с требованиями, но представляет описание и иллюстрирует требования. Модель прецедентов лежит в основе анализа функциональности системы, а значит и в ее будущей работоспособности и пригодности.

Таким образом, при разработке системы были получены следующие результаты, касающиеся модели прецедентов:

- На стадии анализа требований построена модель прецедентов (10 диаграмм, одно описание прецедента).
- На основе модели прецедентов были построены модель архитектуры системы, поведенческая модель, формы пользовательского интерфейса.
- На стадии тестирования проведены 32 базовых системных теста также основанные на модели прецедентов.
- Публикация системы в Интернет планируется в сентябре 2004 г.
- На работу над проектом было потрачено 113 дней или 340 часов рабочего времени.

Список литературы

[1] Кулаков К. А., Сало А. Ю., Ананьин А. В., Крышень М. А. *Web-SynDic: система демонстрации и тестирования синтаксических алгоритмов решения неотрицательных линейных диофантовых уравнений.* Технологии Microsoft в теории и практике программирования. Материалы межвузовского конкурса-конференции студентов и молодых ученых Северо-запада.: Санкт-Петербург, Изд-во СПбГТУ, 2004. С. 43.

[2] Корзун Д.Ж. *Синтаксические алгоритмы решения неотрицательных линейных диофантовых уравнений и их приложение к моделированию структуры нагрузки канала Интернет.* Дисс. на соиск. канд. физ.-мат. наук. : Петрозаводск, ПетрГУ, 2002. 185 с.

[3] Крышень М. А. *Система Web-SynDic: разработка сервера и интерфейса пользователя.* (принято в печать)

[4] Ананьин А. В. *Система Web-SynDic: разработка программного обеспечения на основе прецедентов.* (принято в печать)

[5] Ларман К. *Применение UML и шаблонов проектирования.* : Пер. с англ. : Уч. пос. М.: Издательский дом «Вильямс», 2001. 496 с. ISBN 5-8459-0125-1.

[6] Коналлен Д. *Разработка Web-приложений с использованием UML.* : Пер. с англ. М.: Издательский дом «Вильямс», 2001. 288 с. ISBN 5-8459-0203-7.