Proceedings of the 2013 Federated Conference on
Computer Science and Information Systems pp. 1063–1070

Integrated Model of a Social Navigation System with Self-adaptive Feedback Control Mechanism

Vangel V. Ajanovski
Faculty of Computer Science and Engineering
Saints Cyril and Methodius University, Skopje
ul. Rugjer Boshkovikj 16, 1000 Skopje, Macedonia
Email: vangel.ajanovski@finki.ukim.mk

Abstract—**This paper presents a model of a navigation system in a public information system, that can be used to improve the structure and content of the information repository via self-organization capabilities based on social interaction. This model has the primary goal of establishing a generic and adaptive social-based self-structuring navigation system. To achieve this goal, the model integrates the concepts of social navigation, interaction and self-adaptivity in a feedback control loop. The model gives focus on self-adaptivity and includes elements of social navigation in all parts of the system which enables the implementations based on this model to get social adaptability based on user actions individually, but also as a social environment, in every possible aspect of the functioning of the system. The introduced feedback control loop gives possibility for further autonomous improvements of the organization of the information.**

I. Introduction

THIS paper proposes a model of a navigation system in a information system, that can be used in public knowledge-bases, information portals, news sites, self-support sites, online directories, etc. The model builds towards the enabling of improvement of the structure and content of the repository. We integrate the concepts of social navigation, interaction and self-adaptivity to enable semi autonomous restructuring of the repository.

The original idea was to have as generic approach as possible in order to model a system usable in a wide domain of applications, so some of the concepts are explored from the roots in the past. The concept of social navigation is used as discussed by Dourish and Chalmers in [1], and the generic model is based on the idea of recording interaction on the path that each visitor (navigator) takes, as discussed by Forsberg [2].

There are two main groups of functionalities that are expected from the new navigation system model:

- social interaction should be recorded at all points in the system, either between the system and the visitor or among visitors that are (concurrent or not) visiting the same place of interest in the system – sharing, pointing, recommending and monitoring the published resources and paths through them.
- social interaction history should be used to improve the organization of the navigation and the structure of content – autonomous self-change of the navigation elements should be possible.

In order to include such functionalities, the model first provides the ability to change all navigation point and paths, the whole structure, and then enables social interaction that is associated to the relevant versions of the elements, current to the moment that the interaction has happened. In addition to that, the model enables controlled self-adaptivity, according to results of performed analysis based on social interaction history.

The first step in building this model is the separation of the navigation structure from the content by using a separate navigation sub-system. This sub-system should use a structure that defines (on a conceptual level) and lists (on a logical level) all basic navigational elements that are interesting from a social point of view, and should also include a structure that performs the mapping of logical elements URLs on a physical level. This separation would allow independence of the physical level, which means that changes of the URLs of resources will not have impact on the logical structure and social navigation visible to users. Then, independence on the logical level is introduced, allowing the addition of new elements of social navigation or modification of existing elements, without any change to the physical level.

These are important points for the realization of the structural adaptivity of the system because the constant change of the navigation elements can be the enemy of social navigation (in terms of sharing changed URL, broken paths etc..) For the same reasons, the navigational structure must allow the storage of the entire history of changes (versions of navigation elements) and to handle the shifting of resources in a controlled way that will not disrupt the navigation.

The basic elements of the generic model developed for the case of a web-based public information system are discussed in the following sections.

II. Navigation Elements

The model is organized in levels. At a conceptual level, we identify the following basic types of social navigation elements (with possible extensions of this list), allowing compositions in more complex structures:

- An atomic resource is the basic building element (e.g. a specific resource of this type is: ”Description of the Databases course in the CS study program”, and it can also be an external resource).

- Atomic resources can not exist by themselves, they only exist as a part of resource-sets, which in turn can be:
  - unordered set of logical resources (example: "Guidelines of undergraduate studies at FINKI" or "Course materials for learning Databases"), where the logical resources can be either other sets or atomic elements;
  - ordered set of resources or a directed path (example: "Installation Guide of an Oracle DBMS"), and again, the resource are other sets or atomic elements. The order of the elements is specified manually by the administrator.

At the logical level, the navigational structure of the overall system is built, individual resource sets and their connections are enumerated, and so for each type. For example, the resource "Guidelines of undergraduate studies at FINKI" contains "Description of the CS study program" and "Description of the IT study program". At the physical level, a mapping between atomic resources and physical resources or addresses, is performed. As an example, the logical resource "Description of the CS study program" is mapped to the URL "http://www.finki.ukim.mk/mk/studies/KNI".

Figure 1 shows the model of the navigation structure. The class NavigationElement represents all of the navigational elements that define the logical conceptual level. The type of each element is indicated by the attributes isAtomic and isOrdered. The NavigationElementLink class defines logical links between navigation elements. The Resource class provides the physical mapping of various navigational elements to specific addresses (URL) in the WWW space. Thus, a general graph of all navigation elements and their links is formed. So, this structure can be used to model any web-site or parts of it, or even a "forest" of web-sites.

III. Supporting Self-structuring in the Model

The presented basic conceptual model is then extended to allow self-structuring. For this purpose, two features of the system are required: changeability of the structure and resources. This means that one can change the ordering of resources, or even types of resources with other types, change links between resources (replacing one form to another, adding and deleting logical resources). Also, one should be allowed to change the content of the atomic resources (replacing the mapped physical address to another). Adding an atomic resource (a set of single element) to another atomic resource can produce either an ordered or unordered set, and this is left as a choice for the administrator.

A. Changeability of the Content and Resources

By changeability of resources we actually mean replacing the physical level. The intention is to only change the mappings of logical resources to physical addresses. Also the history of all changes to the mappings should be kept in order to enable later analysis of the behaviour of the users related to each change and pin-point the appropriate version of a resource that is of interest.

The model further elaborates version keeping. In the second row of classes in Figure 1 the versions of the elements and all changes to the physical level are presented. The mapping of physical resources is such that a navigational element is not directly associated with a single physical resource, but in fact a new version of the navigation element is created for any change in the physical mapping. Each change creates a new version object, which records that the version is associated with a change of some resource to a new address. The date of the change is also kept.

In this way, the version entries of each mapping can introduce new versions of the same physical resource – on another date, which would indicate that the navigation element references the same source, again and again, but on different dates. This can interpreted as if the content of the source is changed from time to time, at the version dates, but the URL is still the same. The idea is to have a way to address changes of the content in the model. Also, a completely new version can be created for a new physical resource with a new address. The current version is considered the one with the latest timestamp. The model allows precise monitoring of the changes of the mapping of navigation elements to physical resources, over time.

B. Changeability of the Navigation Structure

The structure of the navigation system can be changed by modifications of the set of navigation links. But, in order to effectively monitor the changes of the structure and the continuity of social navigation – i.e. enabling continuous monitoring of which were original interests of users and where they have migrated after a change, this model only allow for elementary changes and not fundamental change-set where all traces of the past would be gone. The idea is to maintain traceability of each change of the structure.

Only several basic operations are allowed that are implemented in a way that allows monitoring of changes:

- change the type of a logical resource;
- adding new resources to a logical set;
- removing resource from a logical set;
- moving a resource from one to another set.

Considering the way how the logical resources are implemented in the data model using navigational elements – then the allowed set of change operations can in fact completely rearrange navigation structure in any form, when performed as a composition. So in fact any modification can be made but it must be performed gradually, in order to keep proper records of associations of the navigation elements and their version with the social interactions.

1) Changing the type of a resource: Changing a navigation element with another kind is permitted for ordered and unordered sets. As an example, instead of an unordered set of resources, we can decide to have a directed path through the same set of resources or vice versa.

This change is easy to implement over the existing class versions of navigation elements (see Figure 1), with the added ability to record whether a new version includes change of

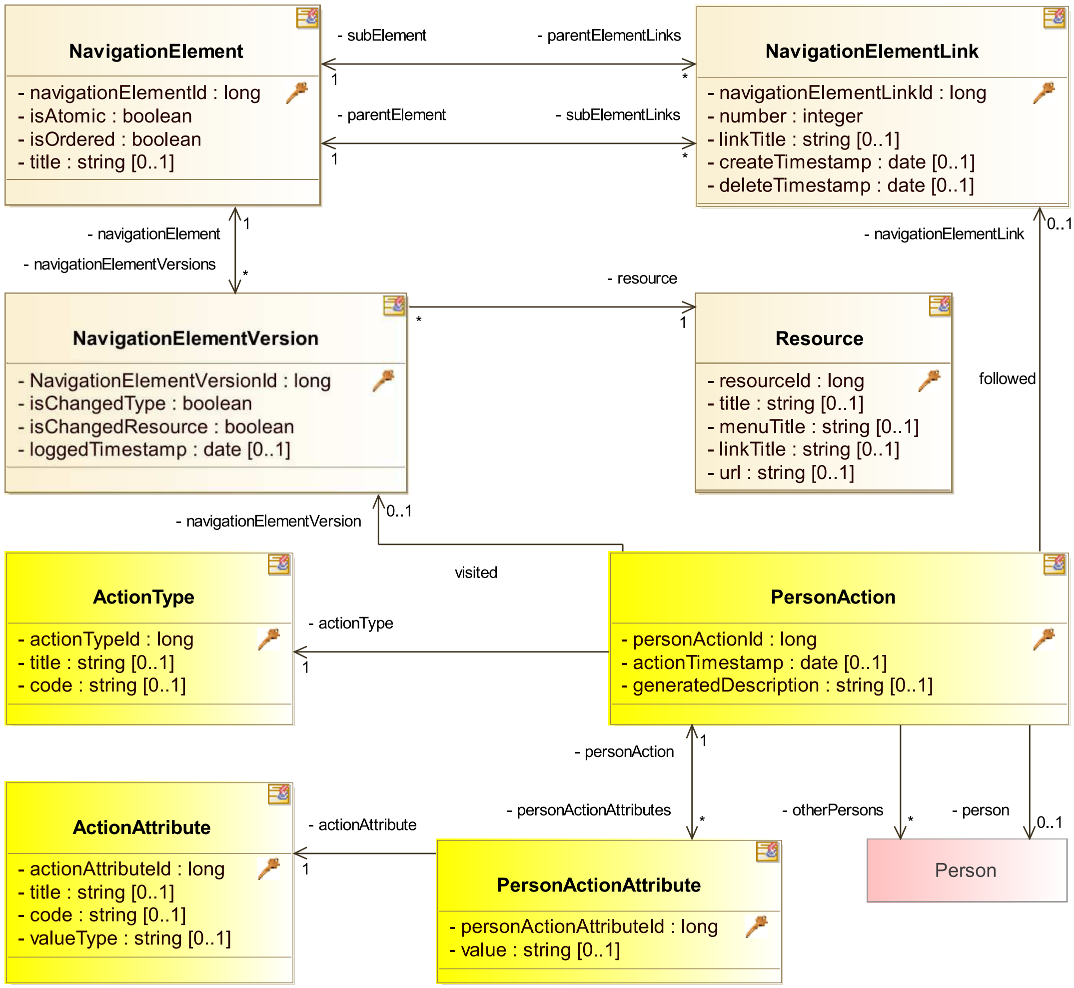

Fig. 1. Data Model of Social Navigation Features.

type. At the same moment, a change of the resource can also be indicated. It is done in this way because it can often be required to reference a new page for the new type and it would be unnatural if even for a small amount of time, first a change of type is visible, and immediately after a change to the mapping to a new content. A change of the element type that does not require a change to the physical resource will be created as a new version and relevant attributes will be denoted, but this new version will still reference the (old) resource. This is done in order to obtain better performance, since search operations are many times more frequent than structural changes.

As discussed at the beginning, an atomic resource itself does not exist, it is always part of a set – in the simplest case it is a set with only one element. For these reasons, a separate consideration of the operation of change of an atomic logical resource into a set, and vice versa, is not necessary. Specific for this case is that what a set has only one element it does not matter if it is ordered or unordered, so the isOrdered attribute is not relevant. So, in this situation the first operation is to replace the type if necessary, and then add new elements or new links where needed and in the required order (especially in the case that the order is important).

2) Adding a logical resource to a set: This operation is realized by creating a new navigation element (if it is a new resource) and then adding a link of the element to the set.

Each link keeps record of the date of creation, which allows one to find out which links were available in a certain period of time.

Note: If the set was made up of only one element, that is atomic, the type of set (isOrdered) has no meaning. So it is ambiguous what will happen once a new element is added. Therefore in this case the administrator must first decide on the new type of the set, perform the change of type operation and only then to execute the operation to add new element.

3) Removal of a logical resource: This operation is realized by setting the value of the attribute deleteTimeStamp in the link that connects the element (child item) to the resource set (parent item). Thus historical data needed for decision making and analytics will not be lost, and navigational elements themselves will continue to exist in the system in case a new reference to them is needed in another place of the navigation structure. Elements that are not longer connected to other elements, are considered orphan elements and can be monitored in case they are needed again in the future.

Note: The re-inclusion of a navigational element that has already been connected in the past, does not change anything in the historical records, but simply creates a new link by performing the operation of adding an existing resource set.

4) Moving a resource from one to another set: This operation is the same as the composition of two operations: remove a resource from the old location set, and add the existing resource to the new location set.

IV. Social Navigation Support in the Model

Social navigation is always the result of interactions of various entities within the system. In order to enable basic support for social navigation throughout the whole of the navigation system, it is required to keep records of all interactions that have occurred, for each interaction type, with each type of navigational element, and to keep additional parameters for each interaction that occurred. This should be kept as a record in time, associated to the current version of the navigation structure. This means that interaction (viewed, read, accessed) should be recorded for the current version of a navigation element and for the link between elements that was interacted with (followed, clicked, etc.).

These requirements are realized through an extension of the previous model, presented in Figure 1. All types of actions that a person can make (indicated by the Person class) are codified according ActionType and are kept in a log of taken actions (class PersonAction), sorted by the action time stamp.

The actions that have a special meaning and are specially recorded in the journal are:

- interaction with other visitors (list of persons to whom the communication is directed);
- interaction with a navigational element (actual version of the element);
- interaction with a navigational link between two elements.

We have defined that some attributes can be of interest to monitor in association to the actions of the visitors. Such attributes and their values can be recorded. Even more, there is no strict list of attributes to be recorded by activity type, but this is left free depending on the requirements of the implementation.

The Person class is not specified in detail, but should be further elaborated in the implementation of the model according to the requirements and can be used from another system. In the case of public information systems where the majority of visitors are people who are not familiar, there are two options for addressing these visits:

- anonymous option – to use only one anonymous object: Person, toward which all actions are recorded;
- weak authentication – for Each new visitor the system creates a new object of the class Person unless there are kept data from a previous visit, such as data kept in cookies or some other technique to determine that a visitor is actually returning to the system. In such case the new actions are added to towards the found previous instance of the Person class.

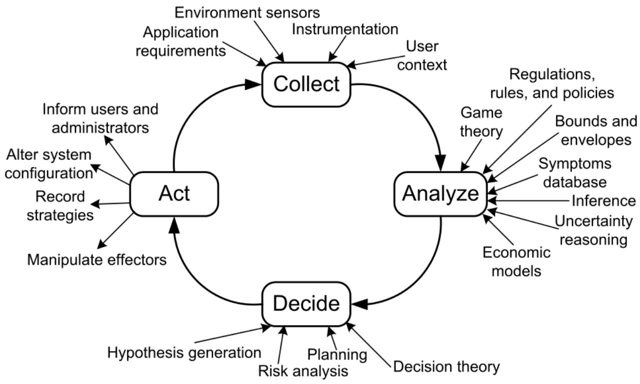

Fig. 2. Feedback control loop[4].

V. Self-adaptivity

The previous discussion identified many elements that allow a constantly changing navigational structure of the system, so it is necessary to monitor all the changes that have occurred and this is necessary for various reasons, the most important being quality control and the ability of the system to constantly adapt to the new needs.

The analysis of the literature in the field of software engineering self-adaptive systems showed different aspects that need to be addressed in such systems, most of which were not relevant for the purposes of social navigation. The most appropriate of all is the model shown in Figure 2, which represents a feedback control loop in a self-adaptive system [3]. This model was originally designed for implementation in terms of communication systems, but can be applied in terms of software engineering self-adaptive software systems [4]. The phases of this cyclic pattern are discussed in the following paragraphs and later the introduction of a control mechanism of that kind is proposed discussed in the generalized model of the navigation system.

In fact this model of a self-adaptive control can be seemed as an appropriate match to the navigation model proposed by Spence [5] (see Figure 3) and the amendments discussed by Riedl [6] for social navigation.

Table I shows the alignment of the concepts of both models. It becomes obvious that one can establish a relation between the two discussed models. The new model uses a feedback control loop to enable self-adaptivity for social navigation in the system.

In fact these are basically two different concepts, one explains the cognitive process of navigation from the perspective of the visitor, and the other describes the process of controlling the operation of a software system that changes its parameters. The idea of linking the two concepts lies in the need of the

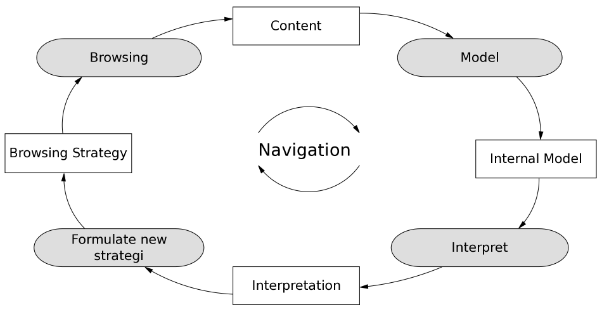

Fig. 3. Navigation Model by Spence[5].

TABLE I
CYCLES OF SOCIAL NAVIGATION AND SELF-ADAPTIVITY.

Social Navigation	Self-adaptivity
review	collecting
modelling	analysis
interpreting	deciding
formulating strategy	action

self-adaptive feedback control loop cycle in order to improve social navigation over the entire set of visitors, but without making direct identification and synchronization of the two processes. The overlap of these two cycles must be understood through an analogy – the mutual coupling of two orthogonally placed gears with varying cogs – the mechanism of social navigation makes many rotations, while the mechanism of self-adaptivity makes one rotation. Through a cycle of the self-adaptive control loop, a number of visitors pass and their (inter)actions are reviewed. The mapping of the cycles in this context would mean:

- viewing information through the system by users generates a multitude of navigational data that some may characterize the overall behaviour of the system, that one needs to collect and analyse;
- symptoms defined by the self-adaptive cycle should apply especially to the navigation system, as they can affect the formation of wrong cognitive model of the visitor or misinterpretation that could lead the visitor leaving the site;
- visitors form their own cognitive model of the system based on how the system is shown and based on personal experience, which can not be directly affected – especially immediately for each user, but one can perform analysis of the behaviour of users, so that in a next cycle assumptions are made about how the visitors perceive the system and what could be changed to improve the navigation
- the interpretation and final formulation of a new strategy for the next search can be influenced by appropriate and timely decisions and actions on behalf of the system, even at low levels - without a major reorganization and just setting some social indicators on well-defended positions, the key in determining whether such interventions help with the navigation process is monitoring the changes made by subsequent self-adaptive cycles in the navigation of the users – what has happened before and after structure changes.

All these features are implementation dependent and are not feasible to be realized in a generic model Therefore it is proposed to plan, define, analyse, design and implement the process of implementation of this generic model with specific issues, depending on the requirements and expectations. What can be generalized is mere conduct the required cycles and their mapping into a generic data model, which will provide the necessary analysis regardless of the implementation at hand. This model is presented in the following text.

A. *Structure for Monitoring Control Cycles*

Because in social navigation systems it is necessary to monitor all activities of visitors and their usage of resource, in order to establishing rules of behaviour of users and help future visitors, and in addition to that the application system needs to have a certain measure of self-adaptivity – which means that that the system has to adapt itself, and not only at the request and parametrization of visitors, there it is necessary to collect enough data for full system operation and association of these two conceptual models. This requires a detailed data model with the following features:

- tracking all actions that occur in a cycle of the self-adaptive control loop;
- tracking the evolution of all system and process parameters across all cycles of the system.

A basic data model is proposed first, for the control cycle and it is shown in Figure 4. This model enable the monitoring of the system through all the cycles. After that this model is linked to the data model for social navigation.

The part of the model that presents the feedback control loop, records are kept on:

- Sets of defined values of the measured systemic and procedural parameters, normal range of values, increased boundaries and critical limits – in order to analyse the status of processes and objects in the system and
- Journal of measured values of parameters, identified symptoms, made decisions and actions that were taken.

The journal is particularly important element because it tracks the success of the control cycle and can be used to find out if the correct symptom was identified, whether requested action at a time was the one really needed, the results can be seen and the history of decisions checked and to conclusions drawn on the correctness of the decision. Of course, some testing would be done manually by the analytical team and some analysis should be programmed for automated monitoring to work.

The phases of the cycle contain elements which seem directly and solely related and ask why it is necessary separation in four phases. The answer is that this form increases modularity – the system is flexible in terms of combining elements. As an example – setting different symptoms for

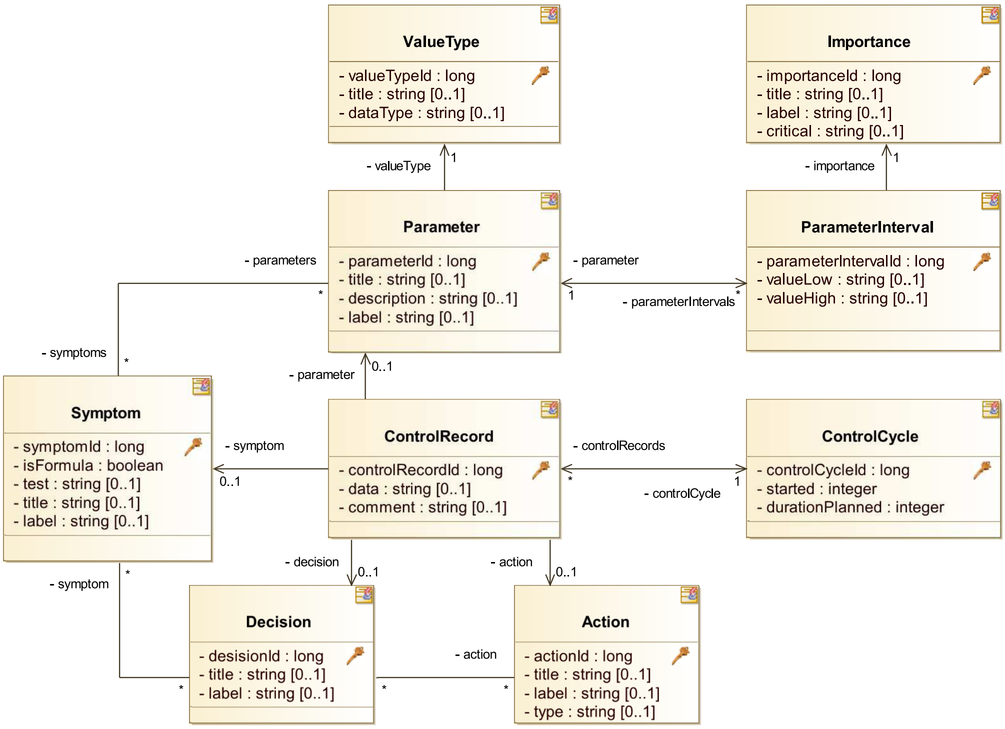

Fig. 4. Data model of the control loop.

the same parameter is allowed, depending on the general context, combination of parameters per symptom, combination of decisions per symptom and combination of actions per decision.

VI. Integration of the Two Models

In order to follow the partial change of the structure of the system and the impact on the behaviour of users, and the impact of changes over the general parameters of the system and boundaries that are allowed, a the system records the changes that have occurred in each self-adaptive cycle.

The model that describes these changes is shown in Figure 5. It should be noted that all associations are optional, which means that the model allows for the incoherent behaviour of the two segments and selecting those components that are required depending on the requirements of the implementation. The RecordedAction class is used to store the information that a certain PersonAction took place within a ControlCycle and is associated with a ControlRecord. Similar to that, the class RecordedModification is used to store the fact that a modification of the structure of content took place and is related to the ControlRecord.

VII. Implementation of the Integrated Model

The steps to implement this generalized model in a production system include final consideration of the requirements of the target system, the amendment of the structures necessary attributes - especially time stamps, recording the users who make administrative changes in the structure and the like.

If the system evolves from the beginning, it is recommended to use this model in all stages as an initial model for defining and navigating through the system.

If the system is implemented as a social and self-adaptive update over the existing system, then you should consider the question of the willingness of the existing system to replace the functionality associated with navigation. If this is not possible, the only solution is the implementation of the navigation model as a separate navigation system, and then mapping and forwarding copies of the shares of users from one system to the other in order to sync the content.

This integrated model can be applied in many different areas, but primarily set in information systems aimed at presenting knowledge and processes related to the management of knowledge and example implementations can be:

- Information portals
- Directories and databases of knowledge systems using
- E-learning
- Social networks oriented learning

VIII. Prototype Implementation and Related Work

The presented model was used to implement a social-navigation based virtual academic adviser[7] that uses recommendations based on social interaction to guide students in the process of course selection, creation of personal study plan until the end of the studies on a longer time scale[8], but also as a self-adaptive control mechanism to create best possible class schedule in term of less conflicts between classes[9]. In

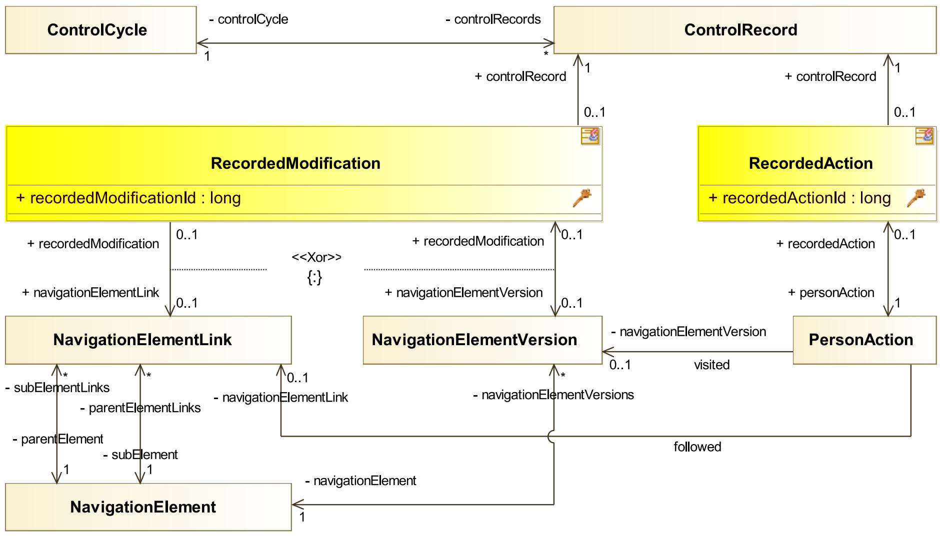

Fig. 5. Integration of the models of social navigation and self-adaptivity.

this implementation, the repository is the database of curricula and study plans for all study programs at a university level.

The virtual academic adviser component gives the student more personal guidance in the process of technical rearrangements of the personal study plan in order to experiment future scenarios with various choices before discussing the final scenario with the real adviser and submitting the application for term enrolment. The first version of the virtual academic adviser enabled the student to try a what if experiment with the various choices on offer such as number of credits per year, choose another study program and specialization profile, rearrange the order of enrolment of courses per future terms etc. After these experiments the student would decide on some preferred scenario and enrol the tern according to the profile and the official rules. In case there were any issues with some scenario, the student could discuss them with the real adviser. It should be noted that the role of the adviser is to give advice and not decide on behalf of the student, and once the choice is legitimate, the adviser can not prevent the student from enrolment, but can only suggest better options.

With the implementation of the new model, the second version of the virtual academic adviser is developed with two additional features: giving the students manual and automated course recommendations and introduction of a new integrated process for term enrolment, class scheduling and construction of timetables. Within this evolution mechanisms for mutual dynamic self-regulation of the processes of term enrolment and class scheduling are investigated. The latest virtual academic adviser is presented in Figure 6.

Each row represents a semester, and each box in the row is a course enrolled in that semester. The semesters are ordered in such a way that the last or active one is on the top, and downwards follow earlier enrollments. Each box shows: the name of the course, whether the lecturer has certified the student was present at majority of lecture hours and is allowed to take exams, the final grade of the student, how many ECTS credits is the course worth if successfully finished. The boxes are colour coded in order to be easily distinguished by the student:

- green boxes represent active courses in current semester
- orange boxes represent courses where the lectures have finished but the student did not have a chance to pass yet
- white boxes represent courses that the student passed
- red boxes represent courses that the student failed.

The system takes into account all interdependencies and course prerequisites and will propose a *realistic plan*. Whether this plan will succeed depends only on the ability of the student to follow and stay through exactly to the new plan and pass all the necessary courses. The system also visually indicates courses that are critical in the future courses where there was a significant amount of failing grades and where students had to re-enroll a course after failing.

The control loop was implemented as a component within the existing information system and used to orchestrate the process of enrollment, phase by phase, student by student or group by group whatever relevant. In the following few paragraphs the four steps of the control loop will are explained with all the included mechanisms.

In the Collect step, the system is monitored and information needed from the students and other system components will be gathered:

- critical time-slots that are almost full
- length of student waiting lists
- numbers of students per status of the enrolment
- numbers of students per year, per group, per status
- number of issues reported by students per category
- number of students without grades for past enrolments

In the Analyse step, the operational status of the system is analysed according to the gathered parameters, historical data and boundary values for several symptoms that are identified

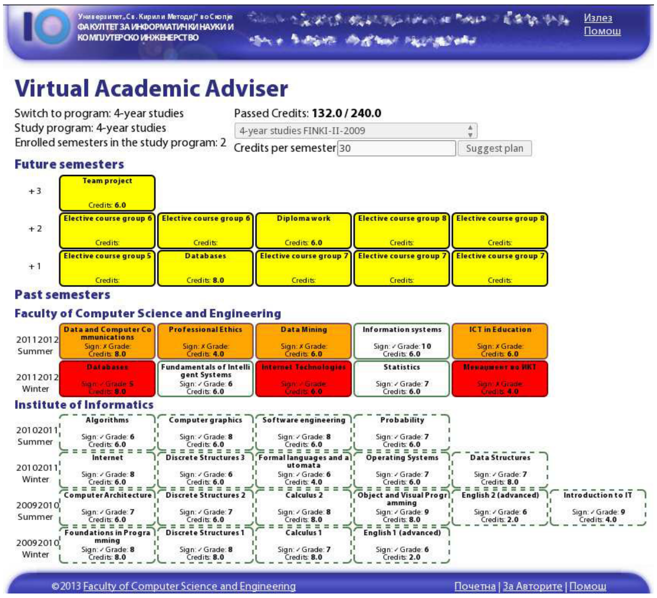

Fig. 6. Integration of the models of social navigation and self-adaptivity.

in the symptom database:

- new groups will be needed soon on a course
- new teachers will be needed soon on a course
- course resources are exhausted
- courses will not be activated due to lack of students
- students ask for courses that are not on offer
- student grades are not input on time
- increasing number of students have complaints

In the Decide step, the system makes decisions on the actions that are to be performed, depending on the severity of the symptomps encountered and how critical are the values of the monitored parameters. In this case mainly the decisions should be made on when and how to act:

- send only information and status via e-mails
- invoke critical alarms to administration staff
- boundary limits can be changed because number of students expected will not exceed significantly
- ask advice on action from administration staff

In the Act step, all actions are orchestrated based on the types of decisions that were made and which simptomps were triggered. The actions that are present are:

- status information is sent to all users
- administration staff is informed that there are issues at hand together with the symptom
- analysis, respective numbers and possible decisions for the decision database
- critical alarms are activated per symptom
- boundary limit are modified and the action is logged
- decisions are logged
- symptoms are logged
- measured parameter values are logged

In such process the course enrolment and time-table creation can be monitored, analysed and acted upon automatically or manually via the proposed control loop framework. In this case it can be argued that the process is successful if finishes on time before the official start of the semester. If that is not the case, the framework gives possibility to monitor the percentage of finished cases of student term and course enrolments and gradually increase the severity of symptoms and frequency of issued critical notifications to the administration staff and to students that have not been active.

IX. Conclusion

The presented model defines a system that is able to change its structure, with traceability of all the modifications of the structure. At the same time records are kept for the interaction of the visitors among themselves and with the system, in association with the exact moment and context regarding how the structure of the system has changed.

This gives possibility to introduce a self-adaptive feedback control loop, that the system will use to monitor itself, identify problems as symptoms and take actions in the form of slight modifications of the structure. The modification are performed in control loop cycles that gives the ability to monitor the causality of the change of behaviour of the visitors and link this change back to a modification in the system, and so investigate if the structural change was an improvement or failure. In such way, the system can undo its steps and take counter measures of bad decisions.

References

[1] P. Dourish and M. Chalmers, "Running out of space: models of information navigation," in *Short paper presented at HCI*, vol. 94, 1994, p. 2326. [Online]. Available: http://fields.eca.ac.uk/deaua/wp-content/uploads/2008/10/hci94-navigation.pdf

[2] J. Forsberg, "Social navigation: an extended definition," *Verfgbar ber: www. nada. kth. se/ forsberg/Documents/(letzter Aufruf am 12.08. 2007)*, 1998. [Online]. Available: http://www.nada.kth.se/~forsberg/Documents/SocNav.pdf

[3] S. Dobson, S. Denazis, A. Fernndez, D. Gati, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli, "A survey of autonomic communications," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 1, no. 2, p. 223259, 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1186782

[4] R. Lemos *et al.*, "Software engineering for self-adaptive systems: A second research roadmap (draft version of may 20, 2011)," Tech. Rep.(October 2010), Tech. Rep., 2011.

[5] R. Spence, "A framework for navigation," *International Journal of Human-Computer Studies*, vol. 51, no. 5, pp. 919–945, Nov. 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1071581999902653

[6] M. O. Riedl, "A computational model and classification framework for social navigation," in *Proceedings of the 6th international conference on Intelligent user interfaces*, ser. IUI '01. New York, NY, USA: ACM, 2001, p. 137144. [Online]. Available: http://doi.acm.org/10.1145/359784.360320

[7] V. Ajanovski, "Towards a virtual academic adviser," in *Proceedings of the 8th International Conference for Informatics and Information Technology (CIIT 2011)*. Molika, Bitola, Macedonia: Institute of Informatics, 2011, p. 146149. [Online]. Available: http://www.getcited.org/pub/103508579

[8] ——, "Personalized adaptive system for term enrollments based on curriculum recommendations and student achievement," in *Proceedings of the International Conference Information Systems 2013*. Lisbon, Portugal: IADIS International Association for the Development of the Information Society Press, Mar. 2013, pp. 342–346.

[9] ——, "Integration of a course enrolment and class timetable scheduling in a student information system," *International Journal of Database Management Systems*, vol. 5, no. 1, pp. 85–95, Feb. 2013. [Online]. Available: http://www.airccse.org/journal/ijdms/papers/5113ijdms07.pdf