DESIGNING AND DEVELOPING SPATIAL DECISION SUPPORT SYSTEMS BY USING ARCGIS ENGINE AND JAVA

Rainer Laudien[1], Sebastian Brocks[1], and Georg Bareth[1]
[1] Dept. of Geography (GIS & RS), University of Cologne, Albertus-Magnus-Platz, 50923 Cologne, Germany
tel.: +49-221-470-6620 fax: +49-221-470-1638
{rlaudien, sebastian.brocks, g.bareth}@uni-koeln.de

ABSTRACT

Modern Spatial Decision Support Systems (SDSSs) combine tools and analysis functionalities of DSS, GIS, remote sensing (RS), and numerical models based on expert knowledge [12]. This paper presents a platform independent SDSS-design and -development approach by using ArcGIS Engine (ESRI®) and Java. With the computer-based SDSSs, the user is able to visualize and analyze (geo-)data and models based on a question-specific decision tree. During the design and development of SDSSs, the diverse knowledge of the potential users needs to be evaluated and considered. A complete SDSS includes that matter of fact and satisfies numerous different decision makers by having user-friendly Graphical User Interfaces (GUI). Thus, the decision maker is able to use these systems in two different ways; on the one hand by using the given default parameters and/or on the other hand by editing the variables based on her/his expert knowledge.

Key Words: ArcGIS Engine, Java, Modeling, Programming, Spatial Decision Support Systems.

1. INTRODUCTION

Nowadays, spatial data, mathematical models and expert knowledge are more and more incorporated in the decision making processes. A modern Spatial Decision Support System (SDSS) combines such data and knowledge within a process-based tool which is developed question-specific according to a given logical decision tree. Turban et al. [23] point out that a SDSS ensures decision support and definite options for management and spatial planning actions. This corresponds with the tasks of Resource Management Information Systems which also serves decision making [17], and of multicriteria decision analysis [15].

Detailed information about the requests, knowledge and personal needs of the potential users are essential during the development of such a computer based system. Therefore, the programming of an innovative SDSS needs to depend on the GIS-, RS-, and model-knowledge of the decision maker.

This study presents the development and implementation of modern SDSSs. The programming of these modular designed systems is based on Java to guarantee platform independent software tools. By the additional use of several different Java based libraries (e. g. ArcGIS Engine) the developed systems meet the requirements of being comprehensive GI-, RS- and model based SDSSs.

The SDSS-design and -programming approach, described in this paper, is being developed for the interdisciplinary research project IMPETUS (An integrated approach to the efficient management of scarce water resources in West Africa). One major task in the third project phase of IMPETUS (2006-2009) is the development, implementation and application of GIS-based SDSSs for the water resource management in selected catchments of Benin and Morocco. The programmed systems will be implemented in a Java/XML based framework [3]. Further information concerning IMPETUS is documented by Speth et al. [22].

2. SPATIAL DECISION SUPPORT SYSTEMS

Before focusing on SDSS it is necessary to point out the term Decision Support System (DSS). The relevant scientific literature does not show a consistent definition of a DSS [13, 21, 23]. However, a DSS can be considered as a computer based system which allows the user to solve semi-structural processes by using comprehensive datasets and analytical models, according to El-Najdawi and Stylianou [2]. This definition points out that the applied resolution methods of a DSS are not trivial at all. In general, a complex DSS is an interdisciplinary solution. Hence, the overall approach of such a system consists in making numerous different methods and analysis available. Therefore, El-Najdawi and Stylianou [2] suggest the need of a similar consideration of data visualization, interpretation and evaluation.

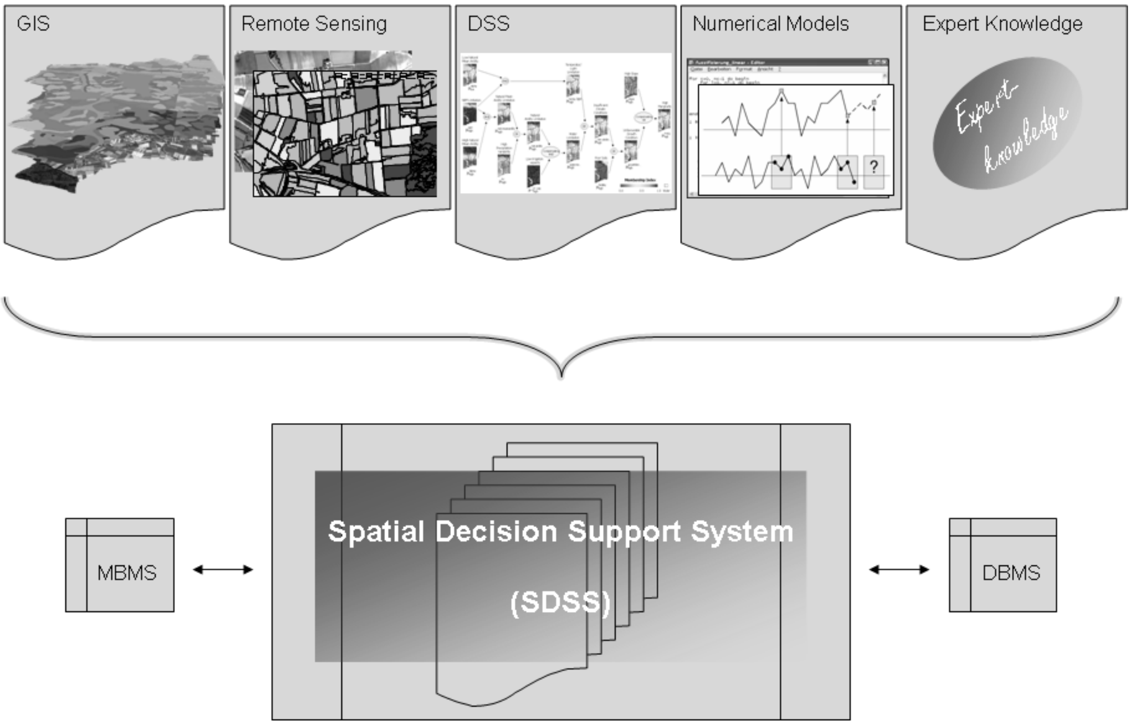

Fig. 1 Configuration of a modern Spatial Decision Support System (modified after [12]).

As soon as spatial data is implemented in a DSS, GIS functionalities get an important role. These functions enable the user to generate spatially diverse decisions. In this context the term Spatial DSS (SDSS) is established in the 1990s [1, 5]. Such SDSSs provide the opportunity to integrate various analytical models, visualize and evaluate the used models and develop management strategies [9, 13, 15, 16, 24].

Modern SDSSs combine functionalities and modules of GIS, DSS, RS, models and expert knowledge [12, 17]. Furthermore, these systems allow the loose coupling of numerical, statistical or knowledge based expert-models, to meet the requirements of being all-inclusive decision support tools [11].

Fig. 1 shows the configuration of a modern SDSS. Consequently, such a system contains data and analysis functions of GIS, RS, DSS, and models and depends on given expert knowledge. To generate a SDSS with these topics, it is essential to implement a Modelbase Management System (MBMS) as well as a Database Management System (DBMS).

3. REQUIREMENTS OF A MODERN SDSS

State-of-the-art computer processors and storage devices are able to process and visualize large amount of data. In addition to that, the calculation of complex model algorithms, raster analyses or work-intensive vector intersections are, in terms of performance, almost no longer relevant. Therefore, the overall requirements of a modern SDSS consist in a variety of several development guidelines:

- Design and development of a user friendly and user oriented GUI
- Implementation of comprehensive (geo-) data
- Implemented warranty of numerous GIS analysis
- Realisation of pixel- and/ or object oriented RS-analysis
- Loose coupling of already developed numerical models
- Development of individual interfaces
- Conception and implementation of a question-specific decision tree
- Integration of expert knowledge during the SDSS run-time

With respect to the approach of developing user-friendly and modular SDSSs, it is essential to regard and minimize potential sources of errors during programming. This means that GUIs, interfaces and DBMSs are implemented which avoid type- and input data errors. Hence, the decision maker is guided through the SDSS in her/his individual adapted view (sight) based on her/his standard of knowledge. Thereby, a user with average familiarity concerning GIS, databases, etc. obtains limited SDSS-functions and -tools while an advanced decision maker gets access to the whole functional range of the system. This differentiation of the user groups comes with different user specific editing possibilities concerning model variables and parameters.

Likewise, all different user specific SDSS versions need to have access to all (geo-) data which are stored in several geo-databases. Implemented database interfaces guarantee the analysis, visualization and modification of the data during the SDSS run-time. In addition to that, comprehensive GIS- and RS-functionalities are applied (e. g. alphanumerical, vector and raster based functions) which are implemented into the Java source code by using ArcGIS Engine.

Furthermore, already programmed numerical, statistical and knowledge based expert models (e. g. coupling IDL, Fortran, Pascal, MS® Excel, etc. based models) which are used for the decision making process are interfaced to the SDSSs. To meet the requirements of generating dynamic and interactive tools, an embedded coupling approach is chosen. This approach comes with the development of Java based model interfaces which connect the SDSSs to the models, transmit parameters and variables, run the models, and transfer the results back from the models to the systems.

To integrate all required decision making processes in one single SDSS, during the software-engineering process, source code needs to be developed specifically based on the SDSS question. The individual decision tree in terms of a float chart helps the software programmer at that stage of the development. Besides specific SDSS analyses which are described above, loops and branches are predominantly implemented to apply the decision tree.

Compared to other, static SDSS approaches the development of modern systems is based on dynamical processes. Hence, an expert is able to implement her/his knowledge during run-time by using e. g. selection-, editing-, or choosing functions. With this opportunity the user is able to change the decision making process interactively. After processing the decision tree, the SDSS results are generated and visualized automatically.

4. DEVELOPMENT OF A MODERN SDSS

For a successful use of the SDSSs after the deployment, it is very important to consider the personal needs and knowledge of the potential users while designing and developing the systems. Therefore, an analysis of needs as well as the definition of the user groups in a preliminary stage of the development is essential. As a lot of the decision-makers are experts in their research topics but usually only have limited experience with computer technology and software concerning GIS and SDSS, the programmed SDSSs need to satisfy both, averaged and advanced users. Therefore, the SDSSs consist of standard as well as complex functionalities. General user specific tools are implemented which generate the decision support based on the decision tree. In addition to that, advanced users who have detailed knowledge in computer models and GIS get the opportunity to apply additional functionalities into the decision making processes. The following sections point out the used development environment, data storage, and the actual programming approach.

4.1 DEVELOPMENT ENVIRONMENT AND DATA STORAGE

Java, an object oriented programming language, is used to develop the SDSSs. The source code is translated into byte code and then executed in a special environment, the so called Java Runtime Environment (JRE). The major part of the JRE is the Java Virtual Machine (Java-VM) which interprets and executes the byte code. By the existence of a Java-VM, the major advantage of programming with Java consists in the fact that all developments run on different computers and different operating systems. Therefore, software which is developed with Java is nearly platform independent [8].

To fulfil the given requests of guaranteeing GIS- and RS-functionalities within the SDSSs, the ESRI® developer library ArcGIS Engine is used to complete the Java source code. With ArcGIS Engine the software developer gets the opportunity to implement spatial analysis functions. The full version of ArcGIS Engine which is accessible by C++, VB.NET, C#, or Java comes with several different extensions which can be integrated based on the user needs. In detail these are: *Spatial extension, 3D extension, Geodatabase Update extension, Network extension, Data Interoperability, Schematics Maplex* and *Tracking*.

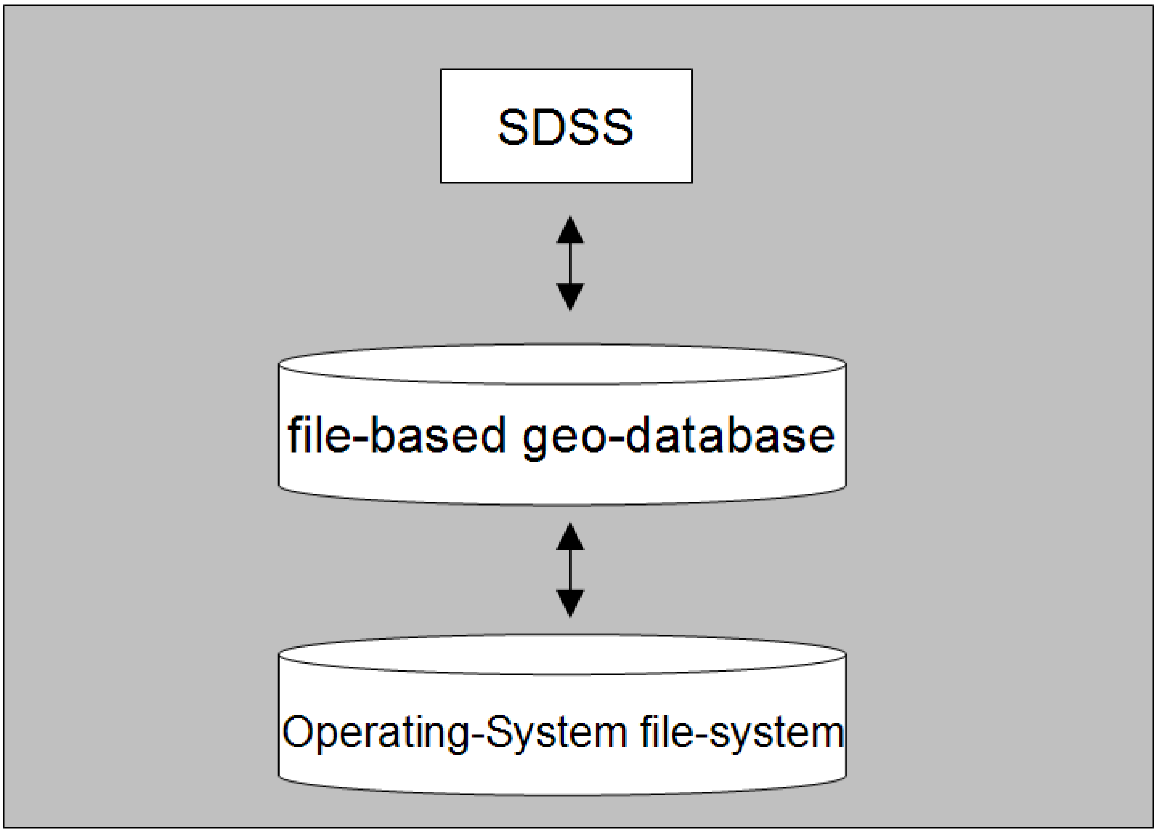

Fig. 2 Data storage based on the ArcGIS Engine file-based geo-databases.

To execute the SDSSs, the JRE and the ArcGIS Engine Runtime Environment need to be installed on the computer. The SDSSs are developed by using the programming environment Eclipse SDK which contains the Eclipse platform (Eclipse 3.2), tools for Java programming and the environment to develop Eclipse plug-ins. In addition to that, Subclipse is used to backup the source code. Subclipse is an Eclipse plug-in that gives the opportunity to connect to a Subversion Repository (SVN). Subversion is a source code repository, which allows several software developers to work on the same project independently without causing backup or versioning errors.

Besides the established Java beans, specific GIS- and RS-components are used for the programming. These are implemented in ArcGIS Engine and can be integrated in the source code easily. Furthermore, coding of additional Java libraries completes the developed SDSS. Apart from that, other libraries, e. g. POI-HSSF components (Java API by Jakarta), are used to couple numerical models (in this context: MS® Excel based models).

The (geo-) data (e. g. raster, vector, and alphanumerical data) of the single SDSSs are stored in file-based geo-databases. Compared to previous geo-databases available with ArcGIS, such as the personal geo-database (ArcGIS Version 9.1), these file-based geo-databases are platform-independent and compressible when used in a read-only context, e. g. when deploying an SDSS to the user who does not write any data to the geo-database. Furthermore, they show a much higher performance when large amounts of data are involved, and these databases are almost not inherent in size limit (the limit is that of the file-system itself). Fig. 2 visualizes the data storage based on the used ArcGIS Engine file-based geo-databases. These geo-databases interact directly with the operating system's file-system. This results in the absent size limit. The large performance boost with large geo-databases results from the poor performance MS Access-based personal geo-databases have once a certain size is reached.

4.2 METHODOLOGICAL APPROACH

Fig. 3 shows the methodological SDSS design and development approach. The figure visualizes a combination between the waterfall approach described by Royce 19] and the one of prototyping [6].

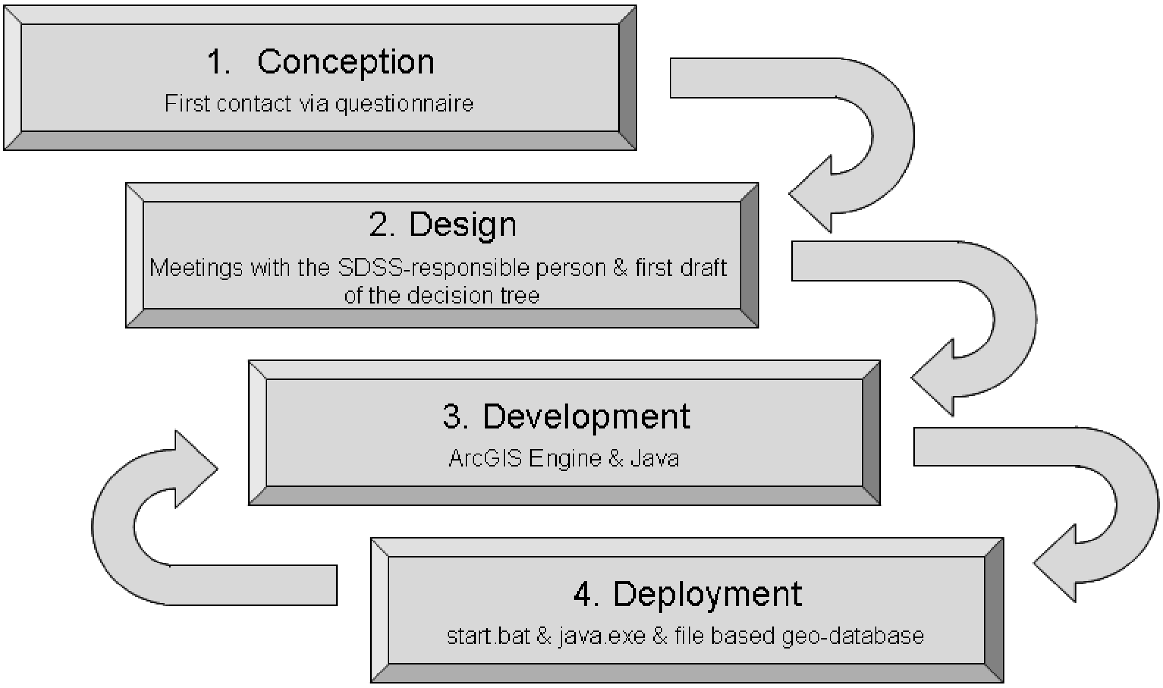
1. Conception
First contact via questionnaire
2. Design
Meetings with the SDSS-responsible person & first draft of the decision tree
3. Development
ArcGIS Engine & Java
4. Deployment
start.bat & java.exe & file based geo-database

Fig. 3 SDSS-design and -development steps: The 4 phase-approach (modified after [12]).

The waterfall model is used in software programming where the development processes can be divided in several phases. Thereby, the single phase commits its result as a binding demand to the next phase. Up to phase three the given approach follows the waterfall model (see Fig. 3). The phases three and four are designed based on a different one, the so called prototyping approach where the requests of the potential users are continuously specified and verified during the development [6]. Therefore, the risk of generating errors is minimized and accidental interactions between the different SDSS modules are early identified. The disadvantage of the prototyping is that the requirements are often not drawn out and documented correctly [6]. Thus, the development process can slow down extensively. This fact results in combining the two approaches (waterfall and prototyping) in a four phase model which allows the development of the SDSSs in a most efficient way.

This combined approach is divided into four development stages. Initially, at the beginning of the SDSS development the overall concept of the system needs to be evaluated with the responsible SDSS-persons (e. g. potential decision makers). This is done by handing out a standardized questionnaire which contain beside others, details regarding the specific question of the SDSS, information concerning the data, and specifications of the potential user-groups (phase 1).

The general design process is embedded in phase 2. During a discussion between the programmer and the SDSS-responsible person, the software realization and the decision tree are developed in terms of a float chart. This visualization shows every single function, algorithm, or data file which will be implemented in the SDSS. In general, this decision tree acts as a detailed task schedule for the software developer. As the SDSS-developer and the expert are designing the system together based on their needs and requirements, this initializing phase is crucial and essential for the success of the whole system.

Phase 2 is followed by the 3rd phase. This stage of the development is characterised by the programming done by the Java programmer. In this development step it is fundamental that the expert and the programmer are in bilateral knowledge transfer to generate a most fitting system (phase 3). After a preliminary first version is implemented, this Beta-SDSS is deployed to the expert for tests and evaluation. This is very important because adjustments can be made easily during this stage of development.

Subsequent to the testing and evaluation of that beta-version, a first version of the SDDS is programmed. This modified 1st version of the system needs to be tested by the potential users. Their feedback and special explanatory notes have to be integrated into later SDSS versions (phase 4). Thus, there is a floating crossover between the phases three and four and they are not completed until the final version of the system is being programmed.

5. SOFTWARE IMPLEMENTATION

Modular and capsulated Java source code is developed based on the programming environment Eclipse SDK (http://www.eclipse.org/). In addition to the SDSS specific requirements, a file-based geo-database is initialized. The database connection is realized by using the *FileGDBWorkspace* Interface of ArcGIS Engine (see the example source-code). By implementing this interface, the SDSS is able to get access to the data during run-time.

Source code excerpt showing the connection to a file-based geo-database.

```
/**
* get the raster named by the parameter from   * Be_E6.gdb"
*/
public IRaster getRas(String aName)
{
  try
  {
    IWorkspaceFactory tWsf = null;
    tWsf = new FileGDBWorkspaceFactory();
    IRasterWorkspaceEx tRws = new
      IRasterWorkspaceExProxy(tWsf.openFromFile
      ("data/Be_E6.gdb", 0));
    RasterDataset tRds = new RasterDataset(
      tRws.openRasterDataset(aName));
    return tRds.createFullRaster();
  }
  catch (Exception e)
  {
    e.printStackTrace();
  }
  return null;
}
```

During the SDSS processing steps the intermediary results are stored in temporary files which are used for further decision stages. The temporary files are stored in an automatically created temporary folder, so that the geo-database is not unnecessarily filled with data. After the decision making process the files of that folder are deleted automatically.

In case numerical, statistical or knowledge based expert models are incorporated in the decision support they are coupled via Java programmed interfaces. Thus, the SDSS accesses the already established model (e. g. IDL, Fortran, Pascal, MS$^{®}$ Excel) by the source code, runs it based on the input parameters given by the user, and presents the model results based on the system requirements. The models are embedded coupled and run SDSS quasi-independent.

Like the development, the software realisations of the SDSSs base on a user-friendly approach. This approach contains a structured user guidance and allows the decision maker to run the system based on the implemented default parameters. Consequently, a dynamic approach is realised which allows the user to adjust the decision support.

The deployment of the SDSSs is done via internet download or CD-ROM. Each developed system contains a specific geo-database, a *.jar file and a *.bat file which can be used to start the SDSS. In addition to that, an ArcGIS Engine Runtime Environment licence is delivered which allows the decision maker to use the embedded GIS- and RS-analysis which are implemented in the developed SDSS.

6. DISCUSSION AND CONCLUSIONS

Modern Spatial Decision Support Systems contain DSS, GIS, remote sensing, model, and functionalities based on implemented expert knowledge [12]. This study presents a methodological approach how such systems can be designed and developed based on ArcGIS Engine and Java. In general, the programming language Java with all its components and libraries acts as an interface between the single modules: GIS, RS, DSS and geo-databases (Fig. 4). By developing Java source code these modules can be connected to meet the requirements of realising a comprehensive question specific SDSS. With this platform-independent programming environment, the Java developer is able to create user friendly GUIs [8]which allow the decision maker either to choose the default expert input variables or customize these SDSS parameters (e. g. model variables) based on his/her knowledge [11]. Thus, the modern SDSSs are highly adjustable based on the individual needs given by the users.

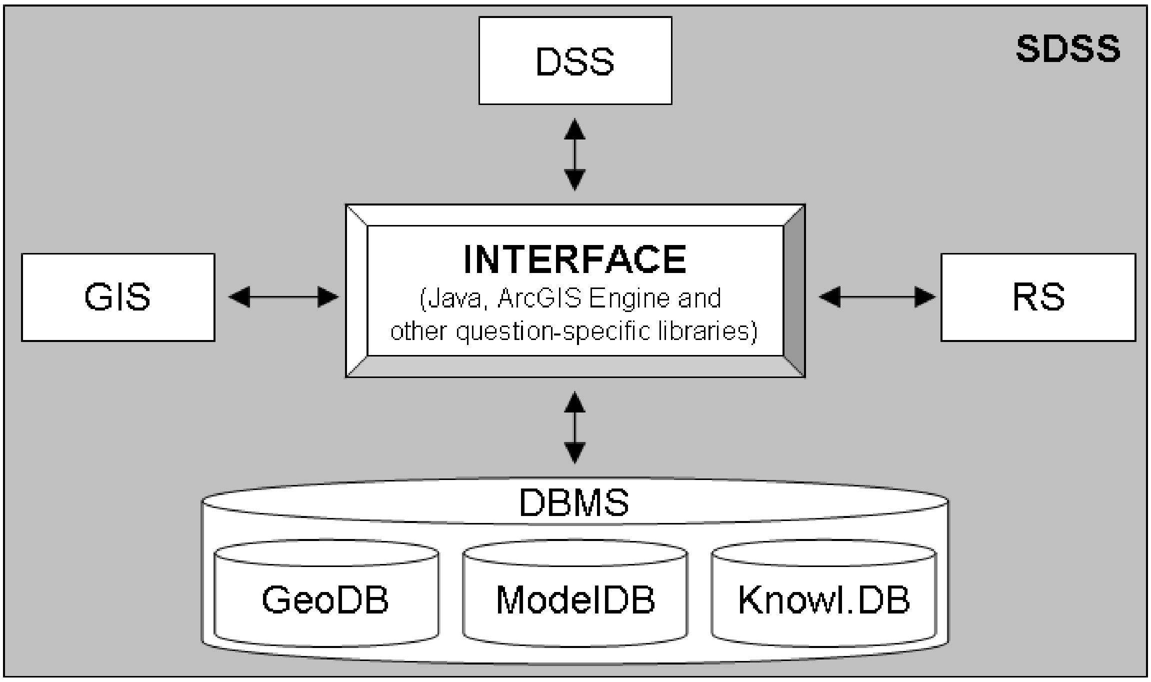

Fig. 4 Components of a modern SDSS as a framework for interfacing DSS, GIS, remote sensing and a comprehensive Database Management System (DBMS).

The importance of interfacing models with GIS are widely discussed in literature, e. g. [7], [14], [18] or [20]. The implementation of the ArcGIS Engine library incorporates an important source of GIS- and RS-tools and -techniques which complement single DSSs with additional spatial references. As the complex decision trees of modern SDSSs are associated with the demand of raster and vector analysis, there is an extensive need to consider such add-ins. In addition to that, numerous other libraries are used to fulfil the individual requests of the single SDSSs, for instance POI-HSSF components (Java API by Jakarta; http://jakarta.apache.org/poi/hssf/index.html) can be implemented to couple MS$^{®}$ Excel based models.

Based on the innovative data storage approach of ERSI’s ArcGIS Version 9.2 geo-data is stored in several file based databases which are connected to the systems. Using file-based geo-databases results in better performance [4], platform-independence, and a much larger maximum file size, compared to the personal geo-databases available with previous versions of ArcGIS (standard with version 9.1). File-based geo-databases work directly with the host operating-system's file-system. In contrast, personal geo-database worked with MS® Access databases, which come with slow performances when a certain data size is reached, and only worked in a MS® Windows environment.

The methods and mechanisms developed and shown in this paper are useful to complement decision making tools. Preliminary SDSS versions presented in [10, 11] indicate that matter of fact. The developed SDSSs are dynamically and interactively accessible. Hence, the decision maker is able to engage the decision support processes during run-time, e. g. by adjusting threshold values for some of the calculations of the SDSS processes. In general, these comprehensive decision support tools are valid for various user groups as they have implemented user-adjusted sights and functionalities.

In conclusion, it should be pointed out, that modern SDSSs can be used as additional tools for modern decision making. They represent complex coherences in terms of multifaceted decision trees within self-contained computer based systems. The developed systems will be implemented in a Java/XML based framework [3]. This framework comes with a centralistic user management system which allows e. g. different SDSS sights for different user groups or an automatic localisation.

ACKNOWLEDGEMENT

This study is part of the interdisciplinary scientific project and is supported by the Federal German Ministry of Education and Research (BMBF) under grant No. 01 LW 06001A and 01 LW 06001B as well as by the Ministry of Innovation, Science, Research and Technology of the federal state of Northrhine-Westfalia under grant No. 313-21200200

REFERENCES

[1] Armstrong, A. P., Densham, P. J.: Database organization strategies for spatial decision support systems. International Journal of Geographical Information Systems (1990), 4(1), 3-20

[2] El-Najdawi, M. K., Stylianou, A. C.: Expert support systems: integrating AI Technologies.-Communications of the ACM 36(2) (1987), 55-65

[3] Enders, A., Laudien, R., Bareth, G., Diekkrüger, B (in prep.): Development of a Java based framework as a platform to implement Spatial Decision Support Systems (2007)

[4] ESRI: What’s New in ArcGIS 9.2. (2007)

http://webhelp.esri.com/arcgisdesktop/9.2/pdf/Whats_New_In_ArcGIS_92.pdf

[5] Goodchild, M., Densham, P.: Initiative 6: Spatial decision support systems (1990-1992). Santa Barbara, CA: National Center for Geographic Information and Analysis (1993)

[6] Gumm, H.P., Sommer, M.: Einführung in die Informatik. 6. Auflage, München: Oldenburg, (2004)

[7] Hartkamp, A.D., White, J.W., and Hoogenboom, G.: Interfacing GIS with agronomic modeling: A Review. Agronomy Journal 91, pp.761-772, (1999)

8. Herter, M., Koos, B.: Java und GIS : Programmierung, Beispiele, Lösungen. Heidelberg: Wichmann, 2006. - IX, 318 S. + 1 CD-ROM, ISBN 3-87907-379-1 (2006)

[9] Keenan, P.: Using a GIS as a DSS Generator. In J. Darzentas, J. S. Darzentas, & T. Spyrou (Eds.), Perspectives on DSS. University of the Aegean, Greece (1996) 33-40

[10] Laudien, R., Giertz, S., Thamm, H.-P., Diekkrüger, B., Bareth, G.: Customizing ArcGIS for spatial decision support - Case study: Locating potential small water reservoirs in Benin. Proceedings of Spie, Geoinformatics 2006, 2006-10-29 – 2006-10-29, Wuhan, China (2006)

[11] Laudien, R., Röhrig, J., Bareth, G., Menz, G.: Spatial Decision Support System zur Modellierung der agrarischen Marginalität in Benin (Afrika). Proc. AGIT'2007, Wichmann, Heidelberg, (2007)

[12] Laudien, R., Bareth, G. (in print): Entwicklung und Programmierung von räumlichen Entscheidungsunterstützungssystemen mit ArcGIS Engine und Java. GISGeoBIT (2007)

[13] Leung, Y.: Intelligent Spatial Decision Support Systems. Springer-Verlag, Berlin (1997)

[14] Longeley, P. A., Googchild, M. F., Maguire, D. J., Rhind, D. W.: Geographic Information Systems and Science., John Wiley and Sons, West Sussex, England, (2005)

[15] Malczewski, J.: GIS and multicriteria decision analysis, Wiley, New York (1999)

[16] Manoli, E., Arampatzis, G., Pissias, E., Xenos, D., Assimacoloulos, D.: Water demand and supply analysis using a spatial decision support system. Global NEST: The International Journal 3(3) (2001) 199-209

[17] McCloy, K. R.: Resource management information systems: remote sensing, GIS and modeling. – Taylor & Francis: Boca Raton, 575 p., (2006)

[18] Miller, I., Knopf, S., and Kossik, R.: Linking general-purpose dynamic simulation models with GIS. - In: D.J. Maguire, M. Batty, M.F. Goodchild (edts): GIS, spatial analysis, and modelling, ESRI Press: Redlands, pp.113-129 (2005).

[19] Royce, W. W.: Managing the development of large software systems. In: IEEE WESCON, Aug. 1970, 1-9 (reprint: Proceedings of the 9th International Conference on Software Engineering, Monterey, CA. (1987), 328-338

[20] Seppelt, R.: Computer-Based Environmental Management, 1. Auflage - September 2003, 278 Seiten, Wiley-VCH, Weinheim (2003)

[21] Singh, A.: Towards decision support models for an ungauged catchment in India, the case of Anas catchment. Dissertation an der Universität Karlsruhe, Fakultät für Bauingenieur-, Geo- und Umweltwissenschaften, Karlsruhe (2004)

[22] Speth, P., Diekkrüger, B., Christoph, M., Jaeger, A.: IMPETUS-West Africa- An integrated approach to the efficient management of scarce water resources in West Africa – Case studies for selected river catchments in different climate zones. DLR– Projektträger im DLR (eds.), GLOWA – German Programme on Global Change in the Hydrological Cycle, Status Report 2005 (2005), 86-94

[23] Turban, E., Aronson, J. E., Linag, T. P.: Decision support systems and intelligent systems, Prentice Hall, New York (2005)

[24] Yeh, A.: Urban planning and GIS. In: P.A. Longley, M.F. Goodchild, D.J. Maguire, and D.W. Rhind (eds.), Geographic information system. Vol. 2, John Willey and Sons, New York (1999)