Adaptation technology of existing decision support systems using Large Language Models

Yevhenii Tolkachenko[1,†], Artem Samokish[1,†], Oleksandr Sychov[1,†], Serhii Toliupa[2,†], Nataliia Gulak[3,†], Elena Dubchak[3,†] and Andrii Maistrenko[4,*,†]

[1]Ivan Kozhedub Kharkiv National University of Air Forces, Sumska Str., 77/79, Kharkiv, 61023, Ukraine
[2]Taras Shevchenko National University of Kyiv, Volodymyrska Str., 60, Kyiv, 03022, Ukraine
[3]Cisco Networking Academy, Hlushkova Ave., 4D, Kyiv, 03127, Ukraine
[4]Scientific Cyber Security Association of Ukraine, Mykhaila Dontsia Str., 2A, Kyiv, 03161, Ukraine

Abstract

The emergence of large language models (LLMs), particularly ChatGPT, has fundamentally transformed user expectations regarding interaction with software, including Decision Support Systems (DSS). While LLMs have introduced intuitive dialogue-based interfaces and the ability to process and generate natural language text, they have also raised concerns over reliability, trust, and stability due to phenomena like "hallucinations." This paper analyzes how these developments have reshaped the requirements for modern DSS and explores strategies for adapting existing systems to meet new expectations. Key requirements now include support for natural language queries and responses, contextual query processing, integration of unstructured data, utilization of open-source information, and adaptive response generation based on user profiles. The paper advocates for a service-oriented architecture (SOA) approach, enabling modular upgrades—such as adding an API layer and interaction services—without altering the DSS's core mathematical or data models. This modular design supports quick adaptation to evolving standards while preserving the DSS's integrity, thus offering a practical pathway to enhance usability, trustworthiness, and relevance in an AI-driven era.

Keywords

Decision Support Systems (DSS), Large Language Models (LLM), DSS adaptation, unstructured information processing

1. Introduction

The emergence of modern large language models (LLM) [1] based on transformers [2, 3], opened up new possibilities for text analyzing and generating. But the real breakthrough, that came to every home, was the emergence of Chat GPT 3 [4]. Search engines like Google made it possible to ask questions in any form, but the answer was a link to a set of resources without data processing. That is, the user must independently analyze the answers. The Chat GPT result was a response in natural language based on a number of sources compilation. It became possible to clarify questions in a dialogue mode. This can truly be considered a new stage in the development of artificial intelligence (AI) systems. Easy access to Chat GPT capabilities allowed a large number of users to test and evaluate its capabilities. Over several years Chat GPT active usage, public opinion began to form regarding of this type systems. Over time, public opinion regarding the LLM capabilities and the risks associated with their use took on certain forms and became a factor influencing the entire AI sector development, including decision support systems (DSS).

CH&CMiGIN'25: Fourth International Conference on Cyber Hygiene & Conflict Management in Global Information Networks, June 20–22, 2025, Kyiv, Ukraine
*Corresponding author.
†These authors contributed equally.
tolkach_g@ukr.net (Y. Tolkachenko); samokisartem@gmail.com (A. Samokish); sychov.for.students@gmail.com (O. Sychov); tolupa@i.ua (S. Toliupa); nataliia.hulak@npp.nau.edu.ua (N. Gulak); 3915922@npp.nau.edu.ua (E. Dubchak); andrii.maistrenko@npp.nau.edu.ua (A. Maistrenko)
0000-0003-3736-2606 (Y. Tolkachenko); 0000-0003-1924-9351 (A. Samokish); 0000-0001-6002-0618 (O. Sychov); 0000-0002-1919-9174 (S. Toliupa); 0009-0000-9584-7113 (N. Gulak); 0000-0001-9739-3960 (E. Dubchak); 0009-0002-1612-9178 (A. Maistrenko)

Using AI in DSS has always been quite natural [5, 6]. Over time, the tasks number, solved by AI in DSS, only increases, as the AI itself capabilities are constantly increasing [7, 8]. Though the AI development and its comprehensive, but incompetent use led to the fact that people began to accumulate mistrust in AI-based systems. Chat GPT had a significant impact on the increase in people's AI mistrust. Chat GPT's strange responses, which were later called "hallucinations", not only "pleased" users, but also increased skepticism towards all AI.

As a result, ordinary users, on the one hand, appreciated the LLM capabilities and expressed a desire to see similar capabilities in modern DSS, on the other hand, users have accumulated AI distrust, which must also be taken into account when developing modern DSS.

However, since there are a large number of high-quality and useful DSS, in addition to the new DSS development, the question arises about the adapting existing DSS possibility to new requirements and the such adaptation feasibility.

This work is devoted to the analysis of how the requirements for modern DSSs have changed and the technology development that will allow adapting the existing DSSs to these requirements.

2. Natural language style queries

2.1. General information

Any DSS that does not support natural language queries is perceived by the user as something outdated. Of course, a DSS should have a graphical interface and an intuitive menu, but users are not only accustomed to the ability to communicate with programs in natural language, but also understand that it is not very difficult.

Everyone understands that it's no longer necessary to independently develop your own converter word2vec [9], your own transformers implementation. Now, the average developer has ready-made libraries from the most famous developers, allowing you to immediately recode a sentence into a numerical vector.

It's even possible to do without direct sentence encoding, and ask questions to LLM, working in the dialogue mode, in which the user's request is contained in a package with additional information. By asking a number of such questions, it's possible to clearly decompose the user's request into simpler formalized instructions for the DSS.

As we can see, in this issue both user expectations and developer capabilities coincide. Therefore, a natural language query becomes the norm for DSS.

It's also necessary to pay attention that most natural language processing systems are designed to work with English. If the DSS must process queries in Ukrainian, it's advisable to use the corresponding LLMs, since when translating text from Ukrainian to English and vice versa, the content may be distorted.

2.2. Generate natural language responses

From technical standpoint, consistent response text generating is no more difficult than request analyzing. The process doesn't look very complicated, but there are a few issues to consider.

The modern user is used to formulating relatively short queries and at the same time wants the system to understand them. Of course, without the query context, it's impossible to give a correct answer. LLMs have taught users that they can clarify their queries, i.e. add new requirements formulated as short queries that do not make sense without analyzing the preliminary conversation. Of course, you can force the user to work with the DSS correctly, but for users accustomed to modern technologies, this causes irritation and, accordingly, reduces the motivation to use such a DSS. Therefore, the accumulation of information that creates the query context is a necessary condition for modern competitive DSS development. Technologies such as Retrieval-Augmented Generation (RAG) [10] allow taking content into account when response forming.

Another context source is the user's personal data. On the one hand, no one wants to share their personal data, no one wants to identify themselves, everyone is about their privacy concerned. But the modern Internet, social networks, and Chat GPT use user information. When an online translator translates a text, it knows the user's preliminary requests and the sites list visited. Accordingly, it knows the user's interest area. Using this knowledge, it makes the translation more accurate. And although people do not want to share information about themselves, they have already gotten used to the fact that all modern services adapt to them. We can say that this is already a modern era requirement, since users expect such behavior from software. Accordingly, the DSS should, if possible, meet these requirements.

Of course, the DSS should not be a full-fledged chatbot that can support a conversation on any topic, but it's still desirable to take into account the context and previous user's requests. In addition, different user groups require different levels of information generalization. For example, users with little experience need more detailed information. Specialists are more focused on facts, and managers tend to receive general information with conclusions. And the modern user expects the DSS to understand over time in what format the answers are needed.

2.3. Using open-source data

Any DSS is based on a models set, and these models don't necessarily have to use AI. The value of DSS models is in their quality and reliability. However, in recent years, users have become accustomed to being able to ask any questions. And it often happens that user questions go beyond the mathematical apparatus, embedded in the DSS. Even 10 years ago, the opinion was established that requests that go beyond the DSS models definition zone should be answered with a refusal. However, at that time, the LLM didn't exist. If we already use the LLM to analyze requests and formulate responses, why can't we use other capabilities? The responses' reliability will usually be much lower, but with the right combination of the DSS mathematical apparatus and the LLM capabilities, additional capabilities can be obtained.

In response to a user query such as "how will a 10% increase in the oil cost affect the rockets cost?" the DSS may refuse to answer, since the rockets cost formula included in its mathematical model doesn't include the oil cost. Or it can use the LLM to determine the impact of oil cost on each of the factors included in the formula for calculating rocket cost, and thus calculate a new cost. Of course, this calculation will be approximate, but it's unknown what exactly the user wants and what accuracy suits him. The DSS task is to warn about the quality of the result obtained. For example, add the phrase "A model of analogies and data from open sources was used in the calculations" to the answer. Thus, the DSS doesn't limit the user, but warns about a decrease in the answer authenticity.

3. Protection of DSS from "hallucinations" emergence

3.1. General information

As is known, "hallucinations" appearance is a disadvantage not only of Chat GPT, but of all LLMs in general. Even if the developers define this as temporary difficulties, it's clear that the "hallucinations" possibility will decrease sharply in the near future, but will not disappear completely. Therefore, the DSS task is to prevent the "hallucinations" appearance in responses. To reduce the "hallucinations" number in the answers, you can reduce the load on LLM. It's necessary to explain in detail what is meant by load.

In fact, the LLM is a very powerful tool with many capabilities. Sometimes DSS developers are tempted to transfer part of their work to the LLM, for example, ask it to calculate a value quality or make a conclusion or description on its own. It isn't very difficult to do - just to form a response request and get a quick result. Exactly such of LLM using leads to the appearance of "hallucinations". Moreover, during DSS testing, there is a high probability of not detecting these "hallucinations". But if the "hallucination" is detected by the user, then the trust in the DSS will be completely violated. If the

DSS has a probability of a "hallucination" appearance at the output, then for the user it's no better than Chat GPT. The user will not figure out where there are more or fewer "hallucinations". For him, the very fact of their possible appearance is a sentence for DSS. Therefore, the issue of LLM correct use is very important.

However, in order to completely prevent the "hallucinations" occurrence, it's necessary to do post-processing and check the responses generated by the LLM. For this, trigrams can be used [11, 12], and conclusions analysis can be carried out using the same scheme described above for queries analysis. But if significant disagreements arise, the response must be regenerated. As we can see, a lot of effort is needed to overcome this problem, and there is always a temptation not to spend so much, arguing that "hallucinations" don't occur very often. However, everyone can ask themselves whether they will use a DSS that sometimes lies?

3.2. Stability

Another problem that occurs when updating the LLM during response generation is that the LLM may give different responses to same requests. It's especially true for LLMs with online access.

Accordingly, if a user received one answer to a request yesterday, and today received a different answer to the same request, this will lead to irritation at the very least, and may also lead to the emergence of entire DSS mistrust. Any user expects that, in the absence of changes, the DSS answers to the same questions will be the same. Otherwise, the user has a feeling of misunderstanding how the DSS works, and accordingly, he stops trusting it.

Unfortunately, when using dynamically updating systems like Chat GPT, additional effort is required to achieve stability.

To solve this problem, you can save user queries, and you can work with software fixed versions. Whatever the case, this problem cannot be avoided.

3.3. Self-improvement of DSS

Users of Google, Chat GPT and other quality systems are accustomed to the fact that they can correct an incorrect answer and the same error will not happen again next time. It's clear that large companies have powerful support departments and improve their systems due to such corrections. It's their development strategy part. But users have already gotten used to such capabilities, and therefore the DSS should also have them. Unfortunately, it's necessary to spend human resources and time to analyze the error and correct the mathematical model. It can't be done quickly. If the DSS uses, for example, neural networks for modeling, then time will be needed for training and testing. The user, in turn, wants his correction to be immediately taken into account by the DSS. To resolve this conflict, the following approach is used - a post-processing module is added to the DSS output, which accumulates and takes into account user feedback. It allows, on the one hand, the user to immediately see the corrected results, and on the other hand, it gives the developers the opportunity to accumulate and analyze user corrections for a certain period, and then make a new DSS modification. From a scientific point of view, this approach can't be called self-training or self-improvement. But for the user, what is important isn't what happens behind the scenes of the user interface, but that the system has taken into account his comments.

When developing such post-processing, there is a need to search for new requests to match added exceptions. For this, it's also possible to use LLM according to the schemes described above. But a large number accumulation of exceptions significantly increases the time and computing resources necessary for processing requests. That's why the SSP should be updated regularly.

3.4. User qualifications accounting

It can be said that "unfortunately, the average level of DSS professional knowledge user has dropped significantly." Or it can be said that "the DSS powerful development the has allowed users to quickly master new areas of activity." Both statements are true and complement each other. Indeed, a modern

user, on the one hand, very easily masters new areas of activity, on the other hand, he lacks basic knowledge and his professional education level isn't very high. Should this be taken into account when developing a DSS? The answer is "definitely yes." The task of modern DSS developers is to do everything possible so that such users can master the work on the DSS as quickly as possible. The use of correct reference, visualization of information, calculations of auxiliary quantities, a user-friendly interface - all this makes mastering the DSS easier and faster. If earlier the presence of such capabilities was an additional plus of the DSS, now it has become a necessary condition for its use. Of course, the development of such capabilities takes a lot of time and resources, while not adding scientific novelty. But a modern DSS can't be targeted at a small group of specialists, which will make it uncompetitive.

4. DSS modification technology

All of the above indicates that modern DSS should have additional properties, but, as we can see, these additional properties don't require changes in either the mathematical apparatus of the DSS or the data structure. Moreover, most of the necessary improvements are the same for certain classes of DSS. Accordingly, it's possible to propose a technology that will accelerate the adaptation of existing DSS to modern requirements. In our opinion, the use of service-oriented architecture (hereinafter SOA, from Service-oriented Architecture) [13, 14, 15] makes it possible to adapt existing DSS to new requirements with minimal effort.

In fact, most of the new properties concern the sphere of interaction between the DSS and the user. Accordingly, if an application programming interface (hereinafter API, from Application Programming Interface) is added to the DSS, then the interaction services with the user and the outside world can use DSS as a service that performs mathematical calculations, modeling and other specialized operations. The service of interaction with users is relatively universal and can be used to work with a whole class of similar DSS.

Accordingly, the DSS modification comes down to adding a module to the DSS that will provide API support.

The services set may vary depending on the tasks. In our opinion, based on the above arguments, the basic additional services set should include:

- Natural Language Query Service;
- Natural Language Postprocessing and Response Generation Service;
- Query context generation service;
- Interaction service with external information sources;
- Unstructured information processing service.

In fact, the DSS also becomes one of the services that can be replaced without changing the structure of service interaction (Figure 1).

The user still has the ability to interact with the DSS through the DSS Graphical User Interface, which was developed specifically for this DSS, and the ability to interact with the DSS through response services appears.

This technology allows adapting the DSS to modern requirements with minimal effort.

It doesn't require changing the programming language in which the DSS is written, nor changing the DSS itself architecture. Adding an API module should not take much time, but it requires effort to understand the DSS itself structure. If a set of additional services has already been created and interaction interfaces have been agreed upon, then in fact, creating and connecting an API module will allow the user to interact with the DSS through the corresponding services.

Compared to modifying the entire DSS, this approach is more expedient and faster.

5. Conclusions

The paper analyzes the impact of LLM emergence on the requirements for modern DSS.

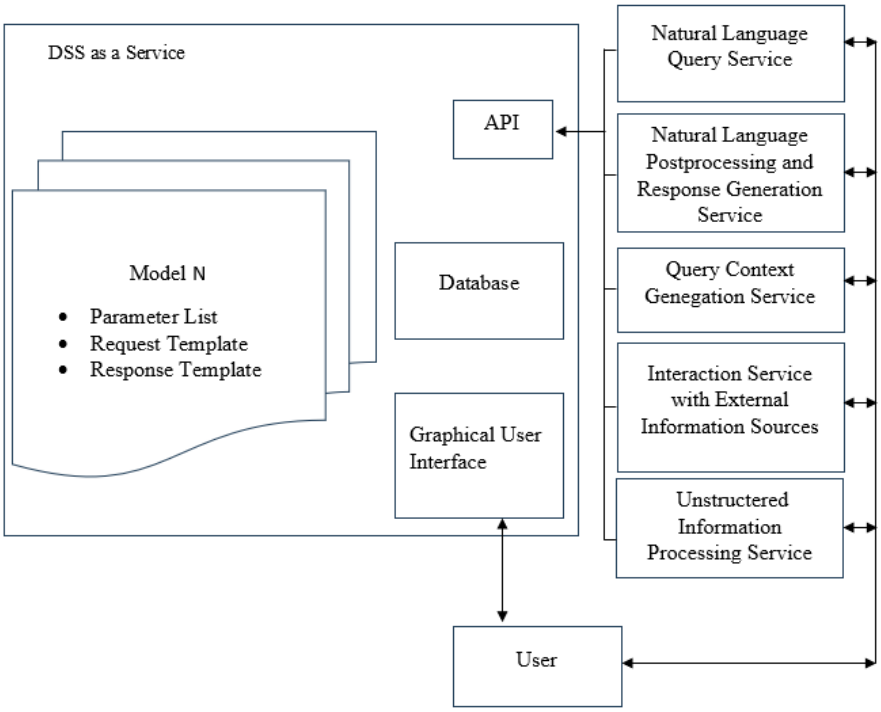

Figure 1: Scheme of interaction between DSS and the user.

The following requirements for DSS are described, which should be taken into account when developing new DSS and improving existing ones:

1. Requirements affected by new capabilities:

- Natural language style queries;
- Responses generation in natural language style;
- Unstructured information processing;
- Data from open source usage;
- Taking into account knowledge about the user (request context).

2. Requirements that take into account the AI shortcomings:

- DSS protection against the "hallucinations" appearance;
- Stability.

3. Requirements that take into account changing of user behavior:

- DSS self-improvement;
- Taking into account user qualifications.

The paper proposes a technology for DSS adapting to modern requirements using a service architecture that ensures the DSS improvement with minimal effort.

By using API, minimal changes are made to the DSS structure. And using of service architecture allows to apply of modules that use LLM to work with different DSSs.

As a result, the user has the opportunity to interact with DSS both through the basic user interface and through additional services that provide support for new functionality that meets modern requirements.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] N. Fathallah, A. Das, S. De Giorgis, A. Poltronieri, P. Haase, L. Kovriguina, NeOn-GPT: A Large Language Model-Powered Pipeline for Ontology Learning, in: Extended Semantic Web Conference 2024, Hersonissos, Greece, 2024.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is All You Need, in: Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017.

[3] M. Banko, E. Brill, Scaling to Very Very Large Corpora for Natural Language Disambiguation, in: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics (ACL '01), Association for Computational Linguistics, Morristown, NJ, USA, 2001, pp. 26–33.

[4] ChatGPT Release Notes, 2024. URL: https://help.openai.com/en/articles/6825453-chatgpt-release-notes, accessed: 2025-07-29.

[5] I. Ostroumov, et al., Modelling and simulation of dme navigation global service volume, Advances in Space Research 68 (2021) 3495–3507. doi:10.1016/j.asr.2021.06.027.

[6] O. Solomentsev, et al., Method of optimal threshold calculation in case of radio equipment maintenance, in: Lecture Notes in Networks and Systems, volume 462, Springer, 2022, pp. 69–79. doi:10.1007/978-981-19-2211-4_6.

[7] O. C. Okoro, et al., Optimization of maintenance task interval of aircraft systems, International Journal of Computer Network and Information Security 14 (2022) 77–89. doi:10.5815/ijcnis.2022.02.07.

[8] V. Larin, et al., Prediction of the final discharge of the UAV battery based on fuzzy logic estimation of information and influencing parameters, in: IEEE 3rd KhPI Week on Advanced Technology (KhPIWeek), 2022, pp. 1–6. doi:10.1109/KhPIWeek57572.2022.9916490.

[9] Y. Goldberg, O. Levy, word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method, 2014. URL: https://arxiv.org/abs/1402.3722.

[10] What is RAG? - Retrieval-Augmented Generation AI Explained - AWS, 2024. URL: https://aws.amazon.com/what-is/rag/, accessed: 2025-07-16.

[11] D. Jurafsky, J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Pearson Prentice Hall, 2009.

[12] N-gram, 2025. URL: https://en.wikipedia.org/wiki/N-gram, accessed: 2025-07-29.

[13] Service-Oriented Architecture Standards - The Open Group, 2025. URL: https://www.opengroup.org/soa, accessed: 2025-07-29.

[14] M. Richards, N. Ford, Fundamentals of Software Architecture: An Engineering Approach, O'Reilly Media, 2020.

[15] M. Brandner, M. Craes, F. Oellermann, O. Zimmermann, Web Services-Oriented Architecture in Production in the Finance Industry, Informatik-Spektrum 27 (2004).