Bloor

White Paper

White Paper by Bloor

Author **Daniel Howard and Robin Bloor**

August 2023

Architecture Patterns and Roadmap for Generative AI in the Enterprise

Generative AI, or just GAI, is the latest computing technology to take both the business world and the general public by storm. Standing on the shoulders of years of developments in data science, machine learning, and deep learning, GAI is a means to generate meaningful, easily digestible outputs from natural language prompts.

Introduction to Generative AI

Generative AI, or just GAI, is the latest computing technology to take both the business world and the general public by storm. Standing on the shoulders of years of developments in data science, machine learning, and deep learning, GAI is a means to generate meaningful, easily digestible outputs from natural language prompts.

Put simply, if you ask a GAI engine to write an article about a given topic, it does so in a few seconds. Ask it to write code for a simple program, and it will write it for you. You ask it a question, and it answers. Well, that's what is intended, but there are pitfalls, as the technology is still in its infancy. Nevertheless, it is obvious why GAI has captured the minds of both business professionals and the computer-using public. The fact that it is highly accessible and easy to use via platforms like OpenAI's ChatGPT or Google's Bard has boosted it significantly.

ChatGPT and other GAI engines are built from Large Language Models or LLMs, which have seen use in niche technology areas for some years but have recently been turned in another direction by the advent of GAI. LLMs are machine learning/predictive models driven by neural networks. Some of them have been trained on enormous datasets of human text, including the entire internet. ChatGPT, for example, has been trained on the whole internet up to September 2021. This is what allows them to recognise and skilfully reproduce patterns of language. They can thus generate masses of meaningful (or sometimes semi-meaningful) text on an endless variety of topics. And so we have the current GAI boom.

There are two broad categories of use cases for GAI within the enterprise.

1. Individual Use

Users can employ GAI to help write articles, take notes, retrieve publicly available information, generate code snippets, answer emails and more. This is directly useful to individual users rather than the business per se, but it will benefit the latter anyway through improved time utilisation.

2. Organisational Use

This concerns the augmentation or even full automation of existing functions and tasks. For example, there are clear applications in customer service, sales support, marketing, content generation, and in more specialised areas, including legal advice and even medical diagnosis. However, there are many areas of application that have yet to emerge. In principle, the possibilities are almost endless. The potential power of GAI makes it difficult to be more specific, at least within the confines of this report. However, if you can provide the LLM with unbridled access to your organisational data, you may increase the potential benefits by an order of magnitude.

However, there are some important considerations to bear in mind if you want to deploy GAI within your organisation for enterprise use cases. The most fundamental is that your LLM (or any other foundational model) will only ever be as good as the data it is trained on. This includes the user inputs it accepts. This makes most of the currently available public LLMs of somewhat dubious use, as they are all trained on the internet, which is:

a) predominantly English-speaking, and hence has a cultural bias

b) frequently incorrect or misleading, and

c) excludes all of the most valuable business and customer data that resides in an enterprise within its private data repositories.

It is thus imperative to maintain the quality and integrity of any custom datasets to which you give it access. Bias detection (and correction) is also very important. Additionally and importantly, compliance regulations, such as GDPR, need to be applied for any data you use to train your LLM, including any personal data. In short, you will need to anonymise any sensitive data that is employed. Real-time data access is another practical possibility: the more up-to-date your LLM is, the wider its area of application will be. Other, somewhat more generic factors, such as integration, performance, scalability, and cost, must also be considered.

Building your Generative AI architecture

Nothing in the applications world is built without architecture. Architecturally, GAI follows a familiar framework: a GAI engine can be thought of as just another application (in this case, the LLM) accessing data (across one or more data stores). The simplest architecture possible, if you can even call it that, is to just use a public LLM without any integration or use of organisational data on your end at all. This could be useful in some scenarios, such as individual staff use or public research, but organisation-level applications will be curtailed by the lack of specific enterprise business data and context. Hence, most companies will need to quickly move to a more robust and sophisticated architecture in order to effectively leverage data within their enterprise. There are two readily available architecture patterns to choose from, which we now describe.

1. Public LLM with RAG

Firstly, you can use a public LLM but enhance it with RAG (Retrieval Augmented Generation). RAG augments your queries by feeding your LLM-relevant, curated, internal enterprise data in order to provide additional context. This means that whenever you query your GAI engine, relevant context derived from within your organisation's authorised data repositories will be added to your query before it is passed through to your LLM and the results generated (this is shown in ***Figure 1***). So your results will be tailored and very specific to your organisation, data, and use case.

This architecture is relatively quick and simple to set up because you do not need to host or re-train your own LLM, and RAG is more cost-effective and efficient than pre-training or fine-tuning foundation models. It is also useful for *"grounding"* your LLM with information that is use-case specific and relevant to your enterprise, enhancing the quality and accuracy of responses and reducing the tendency for your LLM to hallucinate. In short, it is an excellent way to start leveraging GAI, providing some fairly substantial benefits with minimal setup effort.

SIDEBAR 1

The RAG Framework

A RAG framework can be composed of the following five steps, as shown in ***Figure 1***.

1. The Prompt. The user provides a prompt to an application (in this case, a chatbot). The prompt usually contains a brief description of what the user expects in the output.

2. Contextual Search. Additional context is generated with the help of an external program responsible for searching and retrieving relevant information from external data sources. This may include querying a relational database, searching a set of indexed documents based on a keyword, or invoking an API to retrieve data from remote or external data sources.

3. Prompt Augmentation. Once the context is generated, it gets injected into the original prompt in order to augment it. There is now additional, organisation-specific information added to the user's query.

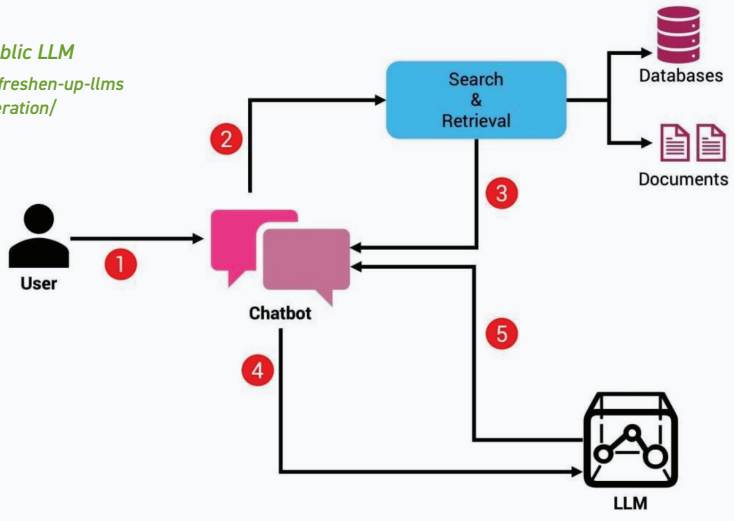
4. Inference. The LLM receives a rich prompt containing both the retrieved context and the original query sent by the user. This significantly increases the model's accuracy because it can now access additional relevant data.

5. Response. The LLM sends the response back to the chatbot, which in turn responds to the user.

Note that the external data used to augment the prompts in RAG can come from diverse sources, including document repositories, databases, and APIs. At the same time, you will need a vector-capable database in order to semantically search through that data and thus provide the context that enables the entire RAG process. Vector databases are discussed further in *Sidebar 2 - What is a vector database?*

Figure 1 –
RAG architecture using a public LLM

Source: https://thenewstack.io/freshen-up-llms-with-retrieval-augmented-generation/



However, the downside to RAG is that it has an intrinsic performance overhead, and there are clear limits to how much RAG can do to contextualise your results. For these reasons, this architecture should be regarded as perhaps an inherently limited first step.

There are also some risks specific to this architecture. Because you are using a public LLM, you have to remember that it is, well, public. This means that when you query it using this architecture, it will train itself on both your original query and any additional context RAG supplies it with. More to the point, it may very well generate responses that are informed by that context to users outside of your organisation. In the worst-case scenario, this could result in data leakage, particularly intellectual property leakage and perhaps even a data breach or data theft by proxy (although to their credit, companies like OpenAI have made several announcements attempting to address these concerns).

Even without anything so explicit, a competitor could, for instance, ask the public LLM about your business and potentially receive results informed by your private information. And, of course, any personal data fed into the LLM using RAG will need to be thoroughly anonymised in order to meet compliance mandates. All of this limits the kind of data you can use to facilitate RAG, making this architecture a reasonable start but not a final destination on your GAI journey.

2. Private LLM with an integrated data layer

The obvious alternative is to simply not use a public LLM and instead fine-tune and maintain a private LLM that lives within your organisation and is tailored to your needs. This clearly demands a more significant initial investment but eliminates many of the security and compliance concerns of a public LLM. It also grants you significantly more control over your LLM, including the ability to actively train it on your own data, enabling much more substantive personalisation and organisation-specific use cases.

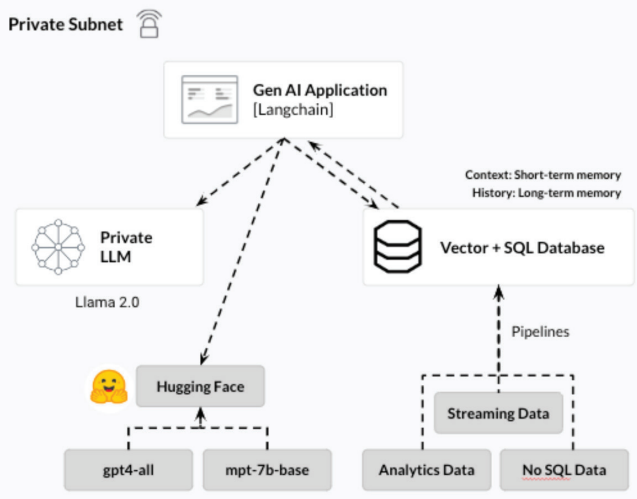
There are several options for private LLMs. First, there are open-source LLMs, such as GPT-3, Llama2.0, Jurassic-1 Jumbo, Alpaca, and MegatronTuring NLG, which can each be downloaded and run on any computer that meets the system requirements. This is a rapidly developing field, so more will definitely emerge. The usual benefits and caveats of open-source apply: on the one hand, there is no upfront cost and plenty of community support; on the other, there is high TCO (Total Cost of Ownership) due to maintenance and other such things and a reliance on community support or the need to develop in-house expertise. Developers can also utilise platforms such as LangChain and Huggingface, which are designed to simplify the application development using LLMs.

Second, several commercial LLMs are available, including Google AI's LaMDA, OpenAI's GPT-J, and Microsoft's Turing NLG. These will have greater upfront costs than their open-source counterparts but will typically offer better performance, dedicated enterprise support, and often an ultimately lower TCO. In terms of differentiation between these different LLMs, the major factor that comes to mind is the size of the data set they have been trained on. In theory, a larger data set may mean better capability, but quality of data is also a factor.

Finally, you could roll your own LLM, perhaps starting from an existing open-source code base. This will be very expensive and represent a significant investment in time and resources, but it provides complete control over your LLM to tailor it to your needs. We expect only the largest companies will seriously consider this option. Microsoft and Facebook are already doing this.

Once you have your LLM, you will need to deploy it as part of an appropriate architecture within your private network. A (fairly generic) example of such an architecture is shown in ***Figure 2***. Note that, as in ***Figure 1***, we have our application (not specifically a chatbot this time), our LLM (now private rather than public) and our data source(s). There are some key differences in this architecture, however. One is that the application directly collects contextual data and funnels it to your LLM, rather than having to go through an external program. Another is that the architecture is entirely situated within your (private) network, as already mentioned.

Figure 2 – Private LLM architecture with an integrated data layer



SIDEBAR 2

What is a vector database?

Vector and vector-compatible databases are designed for storing, searching, and otherwise operating on vector data. Although they have seen a recent surge in popularity because they are well-suited to working with GAI and LLMs, several have been around for quite some time, although vector applications have been fairly niche historically. It is relatively difficult for end users to query or otherwise work with vectors, as they are somewhat complex mathematical objects, but LLMs have no problem interacting with them.

Text and images can easily be stored as vector data through the use of vector embeddings, and indeed, this is the primary way vectors are used to feed information to LLMs. Technically, a vector embedding is a list of floating-point numbers that represent a normalised, multidimensional vector that in turn, represents the original image or piece of text. The distance between two vector embeddings can be calculated just as with any other two vectors, which in this case translates to similarity of meaning. This forms the basis of semantic search for both words and images.

Based on this, it should be quite obvious why vector embeddings, and thus vectors themselves, are so important for building GAI applications and therefore why a vector-capable database can be essential. On that front, there are both specialised vector databases and enterprise-grade databases that support vectors available today, using indexes such as B-Tree, Inverted Index, HNSW, and so on to store vector data. It is also important to note that you will want to search, compare, and otherwise operate on your vector data. This is where the presence (or lack thereof) of various built-in algorithms to this effect can make a difference.

As with any specialty database, you should carefully consider the pros and cons of integrated vector capabilities against those of a standalone vector database before implementing either one. While the latter may offer some performance advantages, for most enterprise use cases adding an additional type of database to your data infrastructure will complicate data integration, and add an extra layer of complexity in general. In addition, speciality vector databases typically provide little to no support for other data types that you might need to build your GAI applications, and may result in the creation of additional data silos (not to mention that it is one more solution to manage). Also, if you want to perform hybrid searches across different kinds of data, it may make sense to find a versatile database that has a vector capability.

The reason vectors are so important is that LLMs work on the basis of vector embeddings, which are essentially a way to represent arbitrary data as multi-dimensional vectors, where the mathematical attributes of the vector signify a sort of machine-readable meaning. To wit, feeding data into an LLM requires you to represent it as a vector embedding.

Vector embeddings are created using an appropriate embedding model (often provided via API). In ***Figure 3***, for instance, we see a visualisation of various vector embeddings created by word2vec, a popular embedding model that embeds words with semantic meaning. This visualisation is limited – it only shows three dimensions for a start – but you can still see that the various words highlighted by the visualisation are grouped loosely together by their semantics.

Using a vector-compatible database to efficiently provide business data to your LLM in the form of vector embeddings, as well as to effectively manage, query, analyse and otherwise operate on your vector embeddings, make a lot of sense. The performance benefits can be substantial, enabling real-time analysis and other advanced capabilities for your vector embeddings, and thus your LLM and GAI engine as a whole.

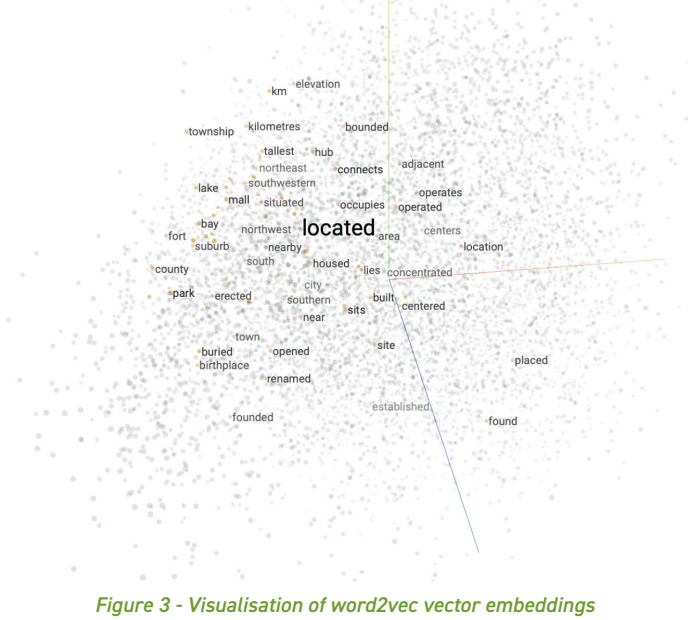


Figure 3 - Visualisation of word2vec vector embeddings

As such, a large number of companies are currently aiming to deploy vector databases specifically to support their GAI engines, implementing an architecture that integrates both LLM and vector databases. Accordingly, vector databases have entered the spotlight alongside (or perhaps in the shadow of) GAI.

Key considerations for selecting a database for Generative AI

By this point in the paper, you should have a good idea of which data infrastructure options are available to you for building out your GAI capabilities, and you know that you will need some kind of LLM backed by some kind of database. However, although we have spent considerable time describing the various types of LLM and their pros and cons, we have not spent much on the qualities one might look for in a suitable database. In this section, we remedy this by providing a list of key features to look for in such a database. In no particular order:

- **Enterprise-grade functionality and scalability.** In the first place, you will want your database to have all of the standard features you would expect from an enterprise product, including the ability to scale out horizontally and proven data availability and data recovery capabilities.
- **Vector capabilities.** We have described the whys of this in detail earlier in the paper, but the short version is that vectors are the building blocks of LLMs and, thus, GAI. You will want to be able to manage them effectively. This includes performing various vector search and nearness operations and algorithms.
- **Multi-model.** You may want to combine your vector data with data from other sources for analysis or whatever else. You may also want to combine multiple other kinds of data for the purpose of, say, training your LLM. Using a multi-model database is not the only way to do this – data integration is also an option, for example – but it might be the most practical.
- **Capable of combining operational and analytical data.** Likewise, you may want to be able to combine operational and analytical data. Perhaps, again, for the purpose of training your LLM.
- **High performance.** Especially in as much as it allows your GAI engine to access and utilise fresh data in real-time, thus making its output as accurate and up-to-date as possible. This is particularly critical when serving use cases that require immediate insights, such as customer-facing conversational AI chatbots and recommendation engines.
- **Flexibility and future-proofing.** We have described several GAI architectures in this paper, and we expect many companies will want to adopt more than one of them at different points in time. Accordingly, you will want to invest in a database that can support the various kinds of GAI architecture.
- **Integration.** Similarly, we would expect an ideal database to integrate with various GAI-oriented tools and libraries, such as OpenAI, LangChain, LlamaIndex and so on, as well as to slot readily into your broader ecosystem.

SIDEBAR 3

GAI in action: potential use cases for LLMs

- **Enhanced Customer Service.**
  Train your LLM on company policy and historic customer interactions, thus allowing it to provide personalised, efficient customer support, resolve common questions or issues and provide up-to-date information to customers.
- **The Sales Process.**
  The LLM can work as a virtual sales assistant, actively engaging the customer by offering detailed product information, making personalised recommendations, upselling, and so on.
- **Order Tracking and Status Updates.**
  Integrate your LLM with your order management and tracking systems, and it could provide real-time updates to order status, shipping information, and delivery times.
- **HR and Employee Support.**
  Just as with customers, an LLM can be used as a virtual assistant for your new employees, helping with onboarding and answering common questions that would otherwise need to go through HR personnel.
- **Data Analytics and Business Intelligence.**
  An LLM can process and analyse large datasets, extract relevant insights, and produce reports that present its findings in a human-readable format, accelerating the business intelligence process.
- **C-Level Advisor.**
  Your LLM could even become a C-level advisor, providing on-demand analysis and advice concerning every area of your organisation's operation.

The future of Generative AI

So far, we have only addressed GAI as it is currently being implemented. We now briefly, turn our attention toward the future of GAI and look to the cutting edge of what only a handful, if any, organisations are currently building with it and where the field might go in the future. Note that due to the forward-thinking nature of this section, our analysis will be significantly more speculative than it was above: it is simply too early to speak in much concrete detail.

Firstly, we anticipate the creation of an LLM-based form of BI (Business Intelligence). We expect this to be double-sided: on the one hand, you could have LLMs driving existing BI systems, as described briefly in *Sidebar 3 - GAI In Action*; on the other, you have to deal with the fact that LLM applications are themselves a new (and evolving) application area, and will therefore be the subject of BI as well. Likewise, you will want a way to monitor and thus govern your LLMs in an appropriate manner, ensuring that they are secure, that they comply with both corporate and government policy, that they are not misusing sensitive data, and so on.

Secondly, it seems unlikely that any one LLM is going to dominate the market (at least, any time soon). In fact, it is entirely plausible that organisations will want to use an ensemble of LLMs to power various different use cases. This could very well include a combination of different types of LLM, including both public and private, open source and commercial, and so on, where each LLM is used in (and trained for) those cases where it provides the greatest utility.

Lastly, there is the integration of GAI with many other forms of enterprise application, including other forms of enterprise AI – recommendation engines, for instance. Several potential forms of this are described in *Sidebar 3 - GAI In Action*, and although we cannot make hard and fast predictions, it is likely that GAI will become a must-have capability for at least some of these purposes.

Note that in all of the above cases, creating and maintaining a robust data architecture for your GAI engine will be essential to realising its full benefits.

Conclusion

The primary aim of this report is to detail the various options for data infrastructure available to prospective adopters of GAI, as well as to provide advice on how, why, and when to go about building each of those infrastructures. It should also be clear that choosing and building an appropriate data infrastructure is an essential part of leveraging GAI, and indeed, that data is the backbone on which GAI is built.

We should note that we do not expect, and in fact would not recommend, most organisations to jump straight to the most complex option described. Rather, we would expect and advise you to start at the beginning with a public LLM – perhaps augmented using RAG – and work your way forward incrementally until you are satisfied that your GAI architecture is meeting your needs. At the outset, a business is unlikely to be able to guess how extensive its use of GAI will eventually become. It needs familiarity, and a gradual approach to adoption will make the time for that.

Just like your GAI engine, your journey to your ideal GAI architecture should be tailored to the needs and desires of your organisation.

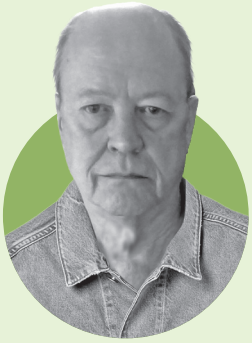
About the authors

DANIEL HOWARD
Senior Analyst

Daniel is an experienced member of the IT industry. In 2014, following the completion of his Master of Mathematics at the University of Bath, he started his career as a software engineer, developer and tester at what was then known as IPL. His work there included all manner of software development and testing, and both Daniel personally and IPL generally were known for the high standard of quality they delivered. In the summer of 2016, Daniel left IPL to work as an analyst for Bloor Research, and the rest is history.

Daniel works primarily in the data space, his interest inherited from his father and colleague, Philip Howard. Even so, his prior role as a software engineer remains with him, and has carried forward into a particular appreciation for the development, DevOps, and testing spaces. This allows him to leverage the technical expertise, insight and 'on-the-ground' perspective garnered through his old life as a developer to good effect.

Outside of work, Daniel enjoys latin and ballroom dancing, board games, skiing, cooking, and playing the guitar.



ROBIN BLOOR
Founder

Robin Bloor is a leading authority and influencer in the industry. In his role as an industry analyst, Robin has become an influential commentator on many corporate IT issues and is in great demand as a presenter at conferences, user groups and seminars addressing audiences across the world on a variety of technology topics from eCommerce through to IT Strategy and trends.

For a decade and a half he was the driving force behind the research effort at Bloor Research, and authored many of its industry reports and product comparisons. He has expertise across the whole field of IT with particular expertise in database, development tools, system management, IT security and hardware technology.

His best-selling business book "The Electronic Bazaar: From the Silk Road to the eRoad" was featured as book of the week by the Times, referred to as "a classic" by Publisher's Weekly (in the US), and described by the Library Journal (also in the US) as "One of the Best Business Books of 2000".

Now living in Austin, Texas, he is still a regular visitor to Europe. He is also a Partner in Hurwitz & Associates, a partner company to Bloor Research which provides IT analysis services to US companies.

He has been influential in shaping the direction and thinking of a generation of IT strategists and continues to provide insight on the direction of IT as it moves forward.

Bloor overview

Technology is enabling rapid business evolution. The opportunities are immense but if you do not adapt then you will not survive. So in the age of Mutable business Evolution is Essential to your success.

We'll show you the future and help you deliver it.

Bloor brings fresh technological thinking to help you navigate complex business situations, converting challenges into new opportunities for real growth, profitability and impact.

We provide actionable strategic insight through our innovative independent technology research, advisory and consulting services. We assist companies throughout their transformation journeys to stay relevant, bringing fresh thinking to complex business situations and turning challenges into new opportunities for real growth and profitability.

For over 25 years, Bloor has assisted companies to intelligently evolve: by embracing technology to adjust their strategies and achieve the best possible outcomes. At Bloor, we will help you challenge assumptions to consistently improve and succeed.

Copyright and disclaimer

Bloor Research International Ltd

20–22 Wenlock Road, London N1 7GU, United Kingdom

+44 (0)1494 291 992

info@Bloorresearch.com

www.Bloorresearch.com