Great Migrations

Support Statement: Overcome the Limits of AI for Large Scale Software Modernization

Watch a podcast of this article.

Video: https://www.youtube.com/watch?v=4I5YKVsJkEY

Q: Why should we use gmStudio instead of just using AI?

A: We believe the optimal solution is to use both. Allow us to elaborate…

There is no question that Artificial Intelligence platforms are bringing important and exciting changes to many fields: art, music, science, business, and computer programming. Technical users of AI are already routinely and productively using AI as a living technical reference for research and to expedite their development work. Even now (mid 2025), this WIKI is already summarized by chatGPT and the other AIs. The application of AI to the field of large-scale software transformation and modernization is particularly important to GM, and we are actively integrating AI into our platform.

Comparing Modernization Tooling: gmStudio + AI versus using AI alone

In our article comparing the tool-assisted rewrite to a more traditional manual-only approach, we describe how teams doing manual upgrades can benefit from also using gmStudio. Likewise, gmStudio can also benefit upgrade teams using AI. Note that gmStudio does not intend to compete with manual development, or AI-assisted development: on the contrary, gmStudio intends complement these activities allowing teams to deliver much more value with less risk and less effort. Some key points on this topic are presented below.

Point	gmStudio + AI	manual-only + AI
Requires an AI properly trained on the detailed information from the legacy code and the required reengineering rules for optimal results	no	yes
Teams may leverage gmStudio to prepare inputs to the AI and integrate outputs generated by the AI	yes	no
Teams may leverage AI to assist with analysis, redesign, and verification	yes	yes
Teams should have development skills in both the source and the target platforms	yes	yes
Provides a flexible, deterministic, precise, and repeatable means of software transformation	yes	unlikely
Typically require powerful hardware or SaaS	no	yes
Can transform a collection of interrelated components as a coherent system	yes	unlikely
May be driven by user-defined transformation rules	yes	unlikely
Offers an extensible library of pre-built transformational rules	yes	unlikely
Can process most large systems in less than 1 minute [1]	yes	unlikely
Automates identifying and removing dead code	yes	unlikely
Automates consolidating redundant code	yes	unlikely
Automates customized COM API replacements	yes	unlikely
Automates customized language transformations	yes	unlikely

Point	gmStudio + AI	manual-only + AI
Automates integrating hand-written code	yes	unlikely
Integrates easily with other DevOps tools	yes	unlikely

[1] gmStudio will typically process up to 30K VB6 or ASP LOC per second.

Integrating gmStudio and AI

The role of AI in tool-assisted software modernization

AI has great potential to assist with the most difficult aspects of software modernization and other large-scale software maintenance projects:

- Analysis to identify opportunities to improve legacy code
- Analysis to identify and evaluate options for redesigning specific code
- Analysis to facilitate developing and executing functional testing procedures

There is no question that AI holds great promise in the area of tool-assisted software reengineering, and teams all over the world researching this important topic. However, the problems are in the details. As I write this in Spring 2024, the application AI for large scale software reengineering still has a long way to go before it reaches its full potential. Here are some thoughts on how integrating gmStudio and AI may benefit large scale modernization work.

Preparing the Legacy System Model

Any significant legacy software system will contain at least 100K lines of code across dozens or hundreds of files. In fact, many of the enterprise systems we see are at least 5-10x this size and sometimes much larger. These system define and use tens of thousands of distinct symbols coming from multiple sources:

- legacy and target computer languages keywords and file conventions
- various internal services and data structures from the application domain, and
- various external services and data structures from the legacy and target platform APIs.

These symbols describe the functionality of the legacy code in a detailed and deterministic way. We assume teams using AI will need to create an LLM and context with these symbols and then use AI to precisely read, interpret, and re-express the legacy functionality on the target platform. This is theoretically possible, but it takes significant expertise and meticulous processes to setup and complete effectively.

The AI model must be "taught" the symbols and the rules to be able to generate meaningful responses about those symbols. The various symbols will require metalanguage parameters that establish their semantics. There will be metadata and rules for the legacy platform, the target platform, and the for the accurate transformation of legacy code into target code. There will also be additional effort to assemble all of the AI generated output into a form that can be productively used built and maintained on the target platform.

In the case of VB6/ASP modernization, we suspect that the ambiguities and anachronisms of old VB6/ASP systems will impair the effectiveness the AI. gmStudio can be used to help prepare the LLM or to generate a .NET version of the legacy system. This information can then be input to the AI for help with analysing, redesigning, and refactoring the .NET code as a second step.

Generating Specialized Inputs and Outputs

Another critical input to AI-assisted software transformation is the user-defined specifications and coding standards for the system once it is rewritten for NET. There are many possible variations of how to write, and rewrite, software systems. The details of the desired coding standards must be accessible to the AI so that it can reliably and repeatedly produce results that follow standards and preserve functionality.

And there is also the task of generating supporting content: csproj files, resx files, config files, AssemblyInfo files, solution files, etc. These artifacts are not fully described by the legacy code; where will they come from? gmStudio is purpose-built to generate these types

of files automatically according to user requirements and this will be helpful to teams using AI.

gmStudio.AI

gmStudio and the Tool-Assisted Rewrite methodology are purpose-built to provide an agile approach to system modernization that dramatically accelerates progress and reduces risk without sacrificing quality or control. The integration of gmStudio with AI provides additional opportunities for even faster delivery and improved technical and functional quality.

GM is using AI in our service delivery workflows where it makes sense. We intend to integrate AI more into our products and methods to flatten the learning curve for new users in 2026.

Please Contact Us to discuss how we can help you bring these two powerful technologies together for the most state of the art modernization project possible.