Youtube link:

https://youtu.be/UYgE1J_ILfE

GNUstep Proposed Enhancement

Liam Beenken: Presenter, Architectural Impact
Cameron Jenkins: Presenter, Alternatives + SAAM Analysis
Evelyn Lee: Leader, Risk Analysis, Testing and Evaluation
Christine Ye: Motivation for Enhancement
Yiting Ma: Sequence Diagrams, Abstract, Introduction, Conclusion/Lesson Learned

Proposal

- Libs-ai
  - New component providing AI functionality
  - Reusable UI components
  - Prewritten AI methods
- ProjectCenter Code Assistant
  - Code autocomplete and suggestions
  - Built into GNUStep IDE

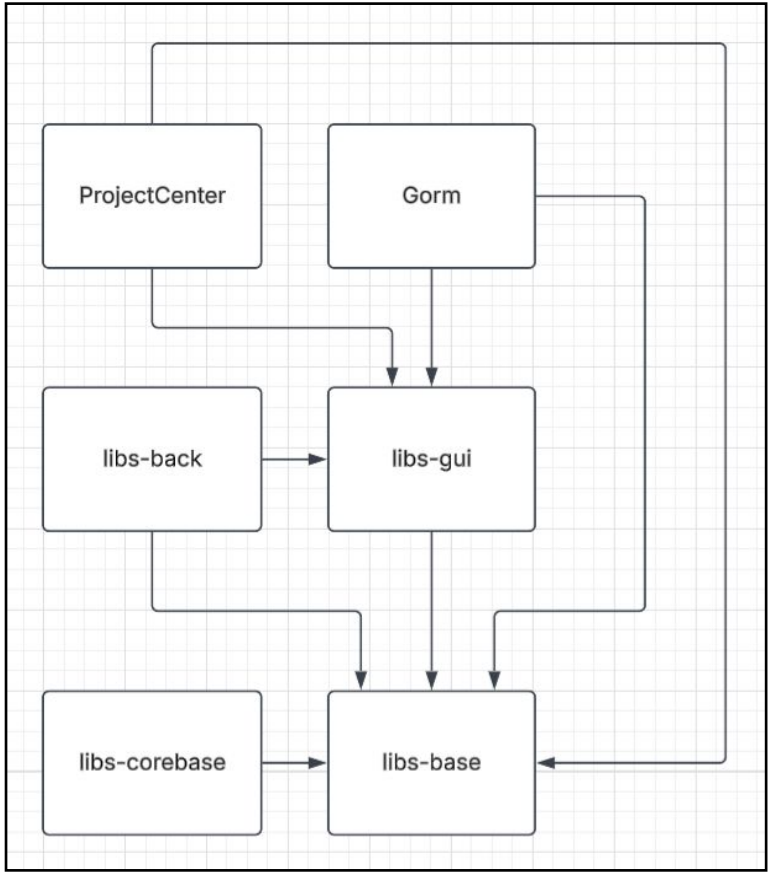
Motivation

- Key NFRs: Capability, Ease of Use, Maintainability
- Code completion tuned to the GNUStep project simplifies development and documentation
- Built in AI support makes it easier for devs to build modern applications
- AI integration gives developers a wider range of tools to create useful software

High Level Architecture Changes

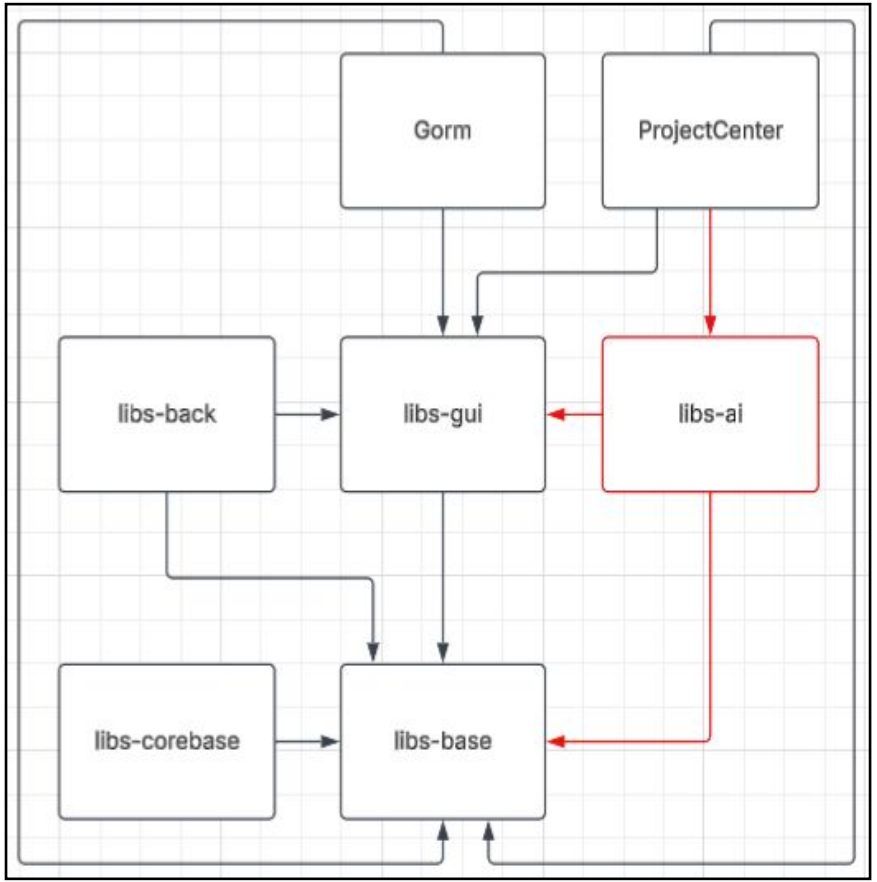
Current Conceptual Architecture

- Layered
- Object Oriented
- Clear Component Hierarchy



Proposed Conceptual Architecture

- New libs-ai component
  - Built on top of libs-base and libs-gui
  - 3 new dependencies
- New ProjectCenter submodule
  - Allows code autocompletion
  - Built on top of libs-ai



Low Level Impact

Libs-ai -> Libs-base dependency

- Libs-ai relies on data structures and methods from libs-base
- Many imports from Source directory will be necessary
- NSURL.m, NSURLRequest.m, NSURLResponse.m for HTTP requests
- NSArray.m, NSString.m, NSDictionary.m, and NSJSONSerialization.m for handling data

Libs-ai -> Libs-gui dependency

- Libs-ai will build AI powered UI components on top of libs-gui
- AIPredictiveTextField would be built on top of libs-gui/Source/NSTextField.m
- AISemanticSearchField would be built on top of libs-gui/Source/NSSearchField.m
- This list of dependencies will increase as more AI components are developed

ProjectCenter -> Libs-ai dependency

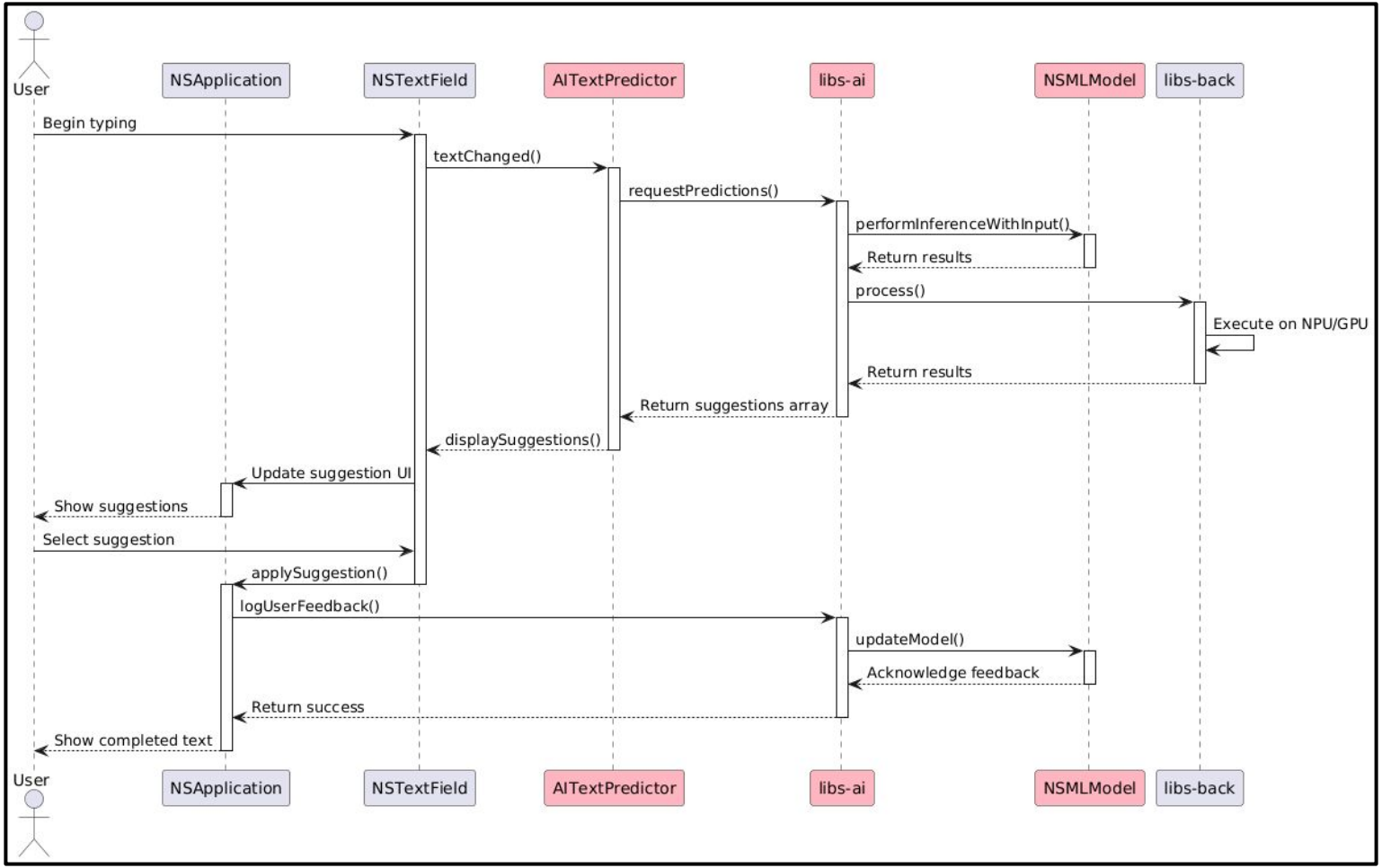
- The code assistant submodule of ProjectCenter will depend on functionality in libs-ai
- Submodule listens for textDidChange() notification from ProjectCenter/PCEditor.m
- Retrieves edited chunk of text
- Calls codeCompletion() method from libs-ai
- Displays suggestion on the editor
- New KeyDown function to accept changes

Testing

Testing Proposal

- Regression Testing to ensure new changes don’t alter existing functionality
- Mock Integration Testing to simulate real world use in user applications
- Stress Testing to ensure concurrent requests don’t overload the AI module
- Fault Tolerance Testing to ensure proper error handling

Use Case Overview


User
NSApplication
NSTextField
AITextPredictor
libs-ai
NSMLModel
libs-back
Begin typing
textChanged()
requestPredictions()
performInferenceWithInput()
Return results
process()
Execute on NPU/GPU
Return results
Return suggestions array
displaySuggestions()
Update suggestion UI
Show suggestions
Select suggestion
applySuggestion()
logUserFeedback()
updateModel()
Acknowledge feedback
Return success
Show completed text
User
NSApplication
NSTextField
AITextPredictor
libs-ai
NSMLModel
libs-back

Approach A - Layered “Plugin-Style” Integration

Overview

- An ai module (libs-ai) acts as a separate layer on top of existing GNUstep libraries. Loaded through plugins/adapter interfaces.
- Minimal changes to libs-base and libs-gui.

Advantages

- Better modularization, keeps ai code separate from the core logic, this makes the system easier to add or change.
- Developers can gradually add AI features without needing to upgrade or adapt immediately.

Disadvantages

- More overhead from the plugin interfaces.
- Not as natively integrated with the core classes.

Approach B - Object-Oriented Extension

Overview

- AI logic embedded directly into existing classes.
- This makes it tightly coupled with libs-gui and libs-base.

Advantages

- Tighter integration, thus likely better performance.
- More seamless integration, methods are built in, no extra step for the plugin.

Disadvantages

- Higher coupling and more difficult debugging.
- Can break older applications that rely on other unmodified classes.

Potential Risk & Limitations

Security and Privacy

- External AI APIs can expose sensitive data.

Maintainability

- AI code complexity can lead to uneven knowledge among developers.

Performance/API evolution

- AI tasks may be GPU or CPU intensive, risking bottlenecks.
- Cloud services may change over time.

Concurrency Effects

Multiple AI requests

- High volumes of AI requests can lead to thread contention or system resource exhaustion.
- Mitigate by using safe data structures, implement queue-based scheduling.

UI responsiveness

- Synchronous or long-running AI operations could block the main UI thread causing the application to freeze/lag.
- Mitigate by offloading interference to background threads or asynchronous tasks.

Scalability

- Handling spikes in AI requests without degrading performance.
- Mitigate by load balancing

Team Issues

Parallel Work Streams

- AI integration vs legacy code refactoring.
- Risk: Merge conflicts, isolated knowledge between team members.

Coordination

- Shared documentation for AI module and existing GNUstep subsystems.

Outcome

- Reduces last-minute integration changes/surprises, encourages knowledge sharing

Lessons Learned

Legacy framework integration

- Mapping modern AI onto older code takes extra effort.

Architectural principles

- Layered or plugin-based designs require maintainability and security.

Documentation and Testing

- Crucial for AI code, thorough coverages allows new developers to better understand what is happening.

Conclusion

Enhanced developer experience

- AI-assisted features reduce coding time and enhance productivity.

Modular and Scalable

- Plugin-style integration enables gradual AI adoption without disrupting GNUstep apps. The OO extension provides tighter integration but can risk higher coupling.

Performance and Security Aware

- Concurrency controls and encrypted cloud APIs ensure responsive and secure AI operations.

Future-Ready

- The libs-ai module supports innovation and can integrate emerging AI services.