基于GAN的唤醒-休眠场景智能模糊测试项目计划书

指导教师：刘琴

团队名称：海底捞针

团队成员：胡宝怡 达思睿 林琪 张诗蔻

1. 引言（Introduction）

1.1 简介

本项目为汽车电子硬件在HIL（Hardware-in-the-Loop）或仿真测试环境中的VCU/域控制器，构建一套专注于“唤醒—握手—Ready—休眠”关键流程的双模式智能模糊测试系统。

系统支持传统模糊测试方式与基于GAN的智能模糊测试方式两种模式，可在统一的测试框架下独立运行或进行结果对比。测试过程遵循多重约束条件（如DBC定义、白名单、禁发规则、CRC校验、速率上限等），并配合自动化注入、异常检测与回灌机制，形成一个可复现、可学习、可优化的智能测试闭环。

最终系统以前后端分离的Web平台呈现，用户可通过浏览器完成从测试计划创建、测试执行与监控、异常分析到结果复现与报告导出的完整流程。

1.2 动机

在传统的随机或规则模糊测试中，在面对“唤醒/休眠”这类强时序、强约束的测试场景时，单一的测试方法存在局限：

- 仅用传统方法：虽能快速覆盖协议边界和常见异常，但难以生成复杂、连贯且“看似合理”的异常序列来触发深层状态机缺陷。
- 仅用GAN方法：可通过学习真实通信分布生成“接近边界”的测试数据，更可能触发复杂时序下的逻辑漏洞，但对数据质量、约束和训练稳定性敏感，在项目初期或数据不足时，其训练成本和不确定性较高。

因此，本项目提供一套兼具两者优势的集成化系统：

- 当需要快速验证、回归测试或进行已知风险点的定向测试时，可使用稳定高效的传统测试引擎。
- 当需要探索未知缺陷、模拟复杂异常场景或提升测试的智能化水平时，可启用GAN测试引擎。

这样既能保持传统模糊的覆盖与鲁棒性，又能利用GAN提升探索性与智能性，从而更高效地定位“误唤醒、误休眠、超时、状态紊乱”等复杂异常问题。为用户提供前所未有的测试灵活性与深

度。

1.3 创新性

- **双模测试架构（Traditional + GAN）：**系统支持两类模糊测试方法在同一框架内运行，可通过参数选择或任务配置自由切换。
- **统一数据约束与验证机制：**无论测试方式来源如何，所有测试用例都经过统一的约束器模块验证，确保合法性与安全性。
- **基于异常的闭环回灌学习：**系统能够识别并记录触发异常的测试用例及其生成方法（传统/GAN），并据此动态调整测试策略。例如，为GAN模型提供“高风险”区域的更高采样权重，或为传统方法动态添加新的变异规则。
- **集成化的Web可视化与分析：**提供对传统方法、GAN方法以及混合策略下测试效果的实时监控与对比分析，支持用户评估不同方法的有效性，并一键导出最有效的异常复现用例。

1.4 主要挑战与应对方法

- **双模型的协调与管理：**两种模糊方式在数据格式、生成逻辑、速率控制上存在差异。应对：设计统一的接口与任务调度层，对两种模式输出进行格式标准化与时序调度。
- **数据口径与安全边界：**确保两种测试方法生成的数据严格符合DBC定义、数值合法范围、白名单/禁发列表及通信速率上限。应对：设计一个统一的、强制执行的约束器模块，所有输入必须通过此模块的校验与修正后才能被注入。
- **GAN训练的稳定性与有效性：**在小样本或特定场景下，GAN训练可能不稳定或模式坍塌。应对：采用结构相对简单（如条件MLP或GRU）的模型，结合早停、正则化等技术，并利用失败重采样和闭环回灌机制来引导和稳定训练过程。
- **与HIL测试平台的集成：**实现与复杂HIL系统的无缝对接存在技术难度。应对：采用分阶段实施策略，首先开发一个虚拟的CAN总线注入器，形成完整的、可演示的系统闭环，积累经验后再逐步对接真实HIL环境。
- **团队对新技术的掌握：**团队成员对HIL测试流程和GAN技术的先前经验有限。应对：前两周大家集中学习，快速补齐基础知识，之后按系统模块进行明确分工，并行推进。

2. 客户需求（Customer Need）

2.1 主要客户与干系人

- 主要客户（第一受众）：测试工程师 / 台架工程师

他们需要在 HIL 或仿真环境中快速创建测试计划、远程控制运行状态、查看实时指标、导出复现脚本与测试报告。

- 次级干系人：测试负责人/技术主管/安全审查人/课程评审人员/后续接手同学。
- 期望体验：用户期望通过一个统一的Web界面，即可一站式完成测试配置、执行、监控和结果分析，能够自由选择并配置“传统测试”或“GAN测试”，并清晰看到不同测试方法所带来的不同测试效果。系统应能显著提升测试的全面性与智能化水平，所有操作应简单直观，结果清晰可比。同时，所有操作与测试结果都应有清晰的日志记录，满足审计与合规性要求。

2.2 用户需求（SMART 用户故事 + 验收标准）

- **测试工程师：**

用户故事：作为一名测试工程师，我希望能在一个界面上灵活选择启用传统测试引擎或GAN测试引擎。对于传统引擎，我能配置变异规则集和强度；对于GAN引擎，我能选择模型版本和采样参数。这样我就能用最适合当前任务的方式，一键启动测试，并获得核心测试指标（如异常触发率、报文受理率）以及排名前N的异常复现脚本。

验收标准：给定用户已登录系统并进入测试计划创建页面，应可以分别看到“传统测试配置”和“GAN测试配置”两个选项，并能独立勾选启用，并在完成必要参数配置后成功启动测试。系统后台则根据其选择，调用相应的引擎生成用例并注入。同时，运行时间窗口内，系统实时看板应持续更新显示异常触发率曲线和报文受理率曲线，并在测试结束后或过程中展示TopN异常列表，且列表中的每一项都提供JSON和CSV格式的复现脚本下载链接。

- **台架工程师：**

用户故事：作为一名台架工程师，我希望在测试任务运行期间，能实时看到当前正在使用的是哪种测试引擎（传统/GAN）生成用例，以及各自产生了多少有效异常，以便我对测试进展有更细致的把握；同时能够通过Web界面随时暂停或紧急停止测试，并且系统能在几秒内响应我的操作，更新状态并在服务端日志留痕记录此次干预，以便我能及时控制测试进程，应对突发情况。

验收标准：给定一个测试任务正在运行中，当用户查看实时监控看板时，界面上除了总体指标外，还应有两个明确标识为“传统测试”和“GAN测试”的独立指标区域或图例，实时更新各自生成的用例数、触发的异常数等；当用户在Web界面上点击了“暂停”或“急停”按钮，那么在5秒内，Web界面上该任务的状态应更新为“已暂停”或“已停止”，并且系统后端日志中应记录下此次用户操作（操作类型、操作时间、用户ID）。

- **测试负责人：**

用户故事：作为一名测试负责人，我希望系统能自动统计并对比同一场景下传统测试与GAN测试的效率与效果并生成报告，包含不同测试方法下关键性能指标（如异常触发率、触发异常的多样性、测试有效性）的差异，并附上一份结论摘要，以便我评估不同测试策略的效果并做出决策。

验收标准：给定一个分别运行了两种测试方法的测试任务完成后，系统生成的总结报告必须包含一个“方法对比”章节。该章节使用图表和数据，清晰展示两种方法在异常触发数量、用例有效率、触发异常的独特性和平均严重等级等方面的差异。

- **安全审查人：**

用户故事：作为一名安全审查人员，我希望能够确认所有测试用例的生成和注入过程都严格遵循了预设的安全约束（如白名单、禁发列表、数值范围、速率限制、校验规则），并且能够查看在整个测试过程中被约束器拦截的测试用例数量及其拦截原因的分布，以便我验证测试过程的安全性与合规性。

验收标准：给定一个测试任务已运行过（无论完成与否），当安全审查人员访问该任务的安全详情面板时，那么面板上应明确显示各项约束规则的启用状态（如“已启用”），并提供一个统计视图，展示“被拦截总条数”以及按“拦截原因”（如“超出范围”、“速率超限”、“在禁发列表”等）分类的分布图或统计表。

3. 项目目标 (Project Goals)

3.1 目标问题

本项目旨在解决汽车VCU（整车控制器）在“唤醒–握手–Ready–休眠”关键流程的HIL测试中，传统模糊测试方法面临的共性难题：

- 测试输入随机性过高，导致无效或冗余用例占比大，测试资源浪费严重；
- 难以系统化、高效率地触及状态机紊乱、时序竞争、边界条件超限等深层逻辑缺陷。

团队计划通过基于GAN的生成方法提升测试用例的针对性，同时保障安全与可复现性。

3.2 用户收益

- **提升测试靶向性**
  - 通过GAN学习正常通信分布特征，生成“看似合理但处于临界状态”的测试序列，直接聚焦潜在故障区域。
  - **用户体验**：减少重复劳动，让测试工程师在相同时间内发现更多有效异常，提升效率感受。
- **加速问题定位**
  - 系统对触发异常的用例进行自动分析和最小化，输出可直接复现的TopN脚本（JSON格式）。
  - **用户体验**：无需在海量日志中手动排查，缩短从发现问题到确认问题的周期。
- **支持测试策略迭代**
  - 提供不同模型版本、采样参数下的测试效果对比（触发率曲线、异常多样性统计），为优化测试计划和模型迭代提供数据依据。
  - **用户体验**：工程师可直观评估策略效果，方便决策和迭代。
- **保障测试过程安全与合规**
  - 前置统一约束器强制执行DBC规范、数值范围、校验规则及通信速率限制，确保测试行为安全可审计。

- **用户体验**：用户无需担心测试会对台架或VCU造成损害，增强操作信心。

3.3 成功度量

- **想法验证对象：**
  本项目的核心构想与解决方案路径，已在项目初期与指导教师刘琴老师进行了多轮讨论，并获得了其原则性认可。同时，团队也咨询了一位在知名汽车零部件企业从事ECU测试的工程师，就传统模糊测试的痛点与本方案的技术可行性交换了意见。外部反馈均认为，将GAN应用于VCU唤醒/休眠流程的测试以提升用例的针对性与效率，是一个值得探索且具有潜力的方向。
- **真实外部客户：**
  本系统的设计主要服务于汽车OEM（主机厂）与Tier 1（一级供应商）的电子电气测试部门。具体而言，我们的目标用户是其团队中的测试工程师与台架工程师。他们日常负责VCU等核心控制器的HIL测试验证，面临着测试周期长、深层缺陷难以触发、问题定位效率低等共同挑战。本系统旨在成为他们手中的一款高效、智能的专项测试工具。
- **客户效益验证方法：**
  为了确认客户是否获得了预期的核心效益，我们将通过以下方式进行跟踪与评估：

1. 定性反馈收集： 在原型或试用阶段，通过深度访谈与结构化问卷，直接向目标客户（测试/台架工程师）了解本系统在易用性、测试效率提升、问题定位辅助等方面的实际体验与主观评价。
2. 系统操作行为分析： 观察并记录客户在实际工作中对本系统的使用频率、依赖程度以及用于核心测试场景的占比，以此间接验证其提供的价值。

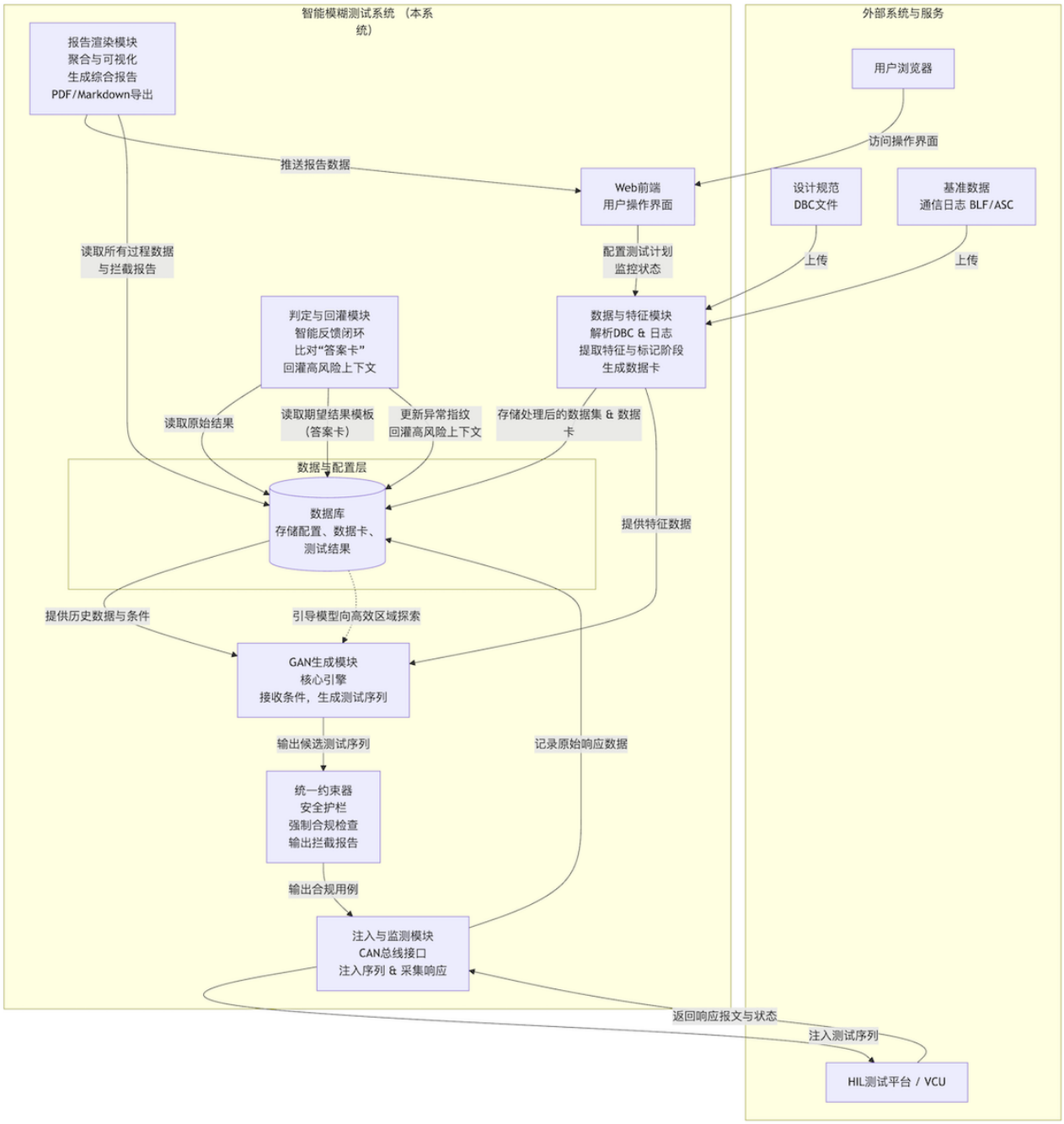
- **客户中心化的成功度量：**
  我们将从测试效能、工程实用性与客户认可三个维度定义成功，具体衡量标准如下：

1. 测试效能提升： 在同等测试资源条件下，相较于传统的基于规则的模糊测试方法，本系统在关键指标上展现出显著优势，例如异常触发率与测试用例有效性的明显提升。
2. 问题复现与洞察能力： 系统能够高效地复现所发现的疑似缺陷，并提供清晰的复现路径与上下文信息，同时具备发现现有测试用例库未能覆盖的新状态或潜在风险的能力。
3. 交付物价值认可： 项目最终产出的测试报告、复现脚本、模型分析等核心交付物，其专业性、实用性及对实际测试工作的指导意义，能够获得导师或潜在企业用户的原则性认可与积极评价。

4. 系统描述 (System Description)

4.1 系统边界与交互

本系统在技术上划分为一个前后端分离的Web应用。后端服务通过REST API与WebSocket与前端进行配置下发与状态同步，并通过标准化接口与客户侧的HIL测试平台或虚拟仿真环境进行指令注入与数据采集。



如上图所示，系统的边界与数据交互如下：

- 系统的输入包括：
  - 设计规范： 由用户通过Web前端上传的车辆网络DBC描述文件。
  - 基准数据： 由用户上传的、捕获自正常工况的VCU通信日志（BLF/ASC格式）。
  - 测试策略： 用户通过Web前端配置的，以JSON格式定义的约束条件、采样参数与任务目标。
- 系统的输出包括：
  - 测试报告： 由报告渲染模块生成的，涵盖执行概览、效果对比、拦截统计的PDF/Markdown文档。
  - 工程资产： TopN最小复现脚本（JSON）、新发现的异常状态指纹。
  - 过程记录： 完整的系统运行日志与所有测试用例的元数据，确保过程可回溯。

- 外部交互实体：
  - 用户浏览器： 用户与系统交互的唯一入口，通过Web前端进行操作。
  - HIL测试平台 / VCU： 系统的物理测试对象与环境，接收注入的测试序列并返回响应数据。

4.2 核心工作流程

系统的工作流程遵循一个清晰的智能测试闭环：

- 准备与配置： 用户通过Web前端上传DBC文件和基准日志，并创建测试计划。这些配置信息驱动数据与特征模块进行数据解析和特征提取，为测试准备好数据基础。
- 智能生成与安全校验： GAN生成模块作为核心引擎，根据接收到的条件生成候选测试序列。所有生成的序列必须立即经过统一约束器的安全校验，确保其符合所有预设规范，充当系统的“安全护栏”。
- 注入与监测： 通过校验的合规测试用例由注入与监测模块按既定节奏注入到CAN总线，并同步监测HIL测试平台及VCU的响应。
- 判定与反馈学习： 判定与回灌模块将监测到的实际响应与预期行为进行比对，判定测试结果。最关键的是，成功触发异常的用例其上下文特征会被记录，并作为反馈信号“回灌”至GAN生成模块，引导其后续向更高效的测试区域探索，实现自适应优化。
- 报告与可视化： 整个过程中的所有数据被报告渲染模块聚合，生成综合测试报告，并通过Web前端实时展示给用户。

4.3 主要模块构成

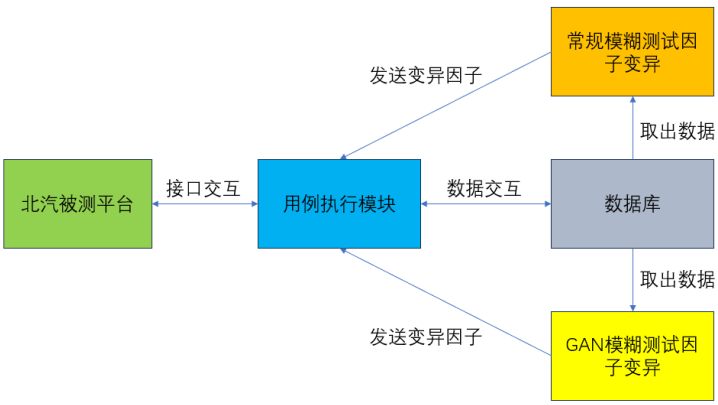
- 数据与特征模块： 作为测试任务的数据基础。该模块负责解析DBC文件以获取信号矩阵，并读取正常的通信日志（BLF/ASC格式），从中提取“唤醒/休眠”场景下的典型信号序列、时序特征（如阶段间延时Δt）并进行阶段标记。输出结果为用于模型训练、验证和测试的数据集，同时生成描述数据来源与处理版本的数据卡。
- GAN生成模块： 本系统的核心引擎，负责生产测试用例。它接收流程控制模块下发的条件（如当前测试阶段、目标ID、关键信号状态），结合采样温度与随机种子，通过生成器网络输出候选测试序列。该模块严格遵循“纯生成”原则，其输出不经过任何传统的、基于规则的变异算子。
- 统一约束器： 充当系统的安全护栏，对所有出站测试序列进行强制性合规检查。检查规则基于DBC文件与JSON策略配置，包括信号白名单、功能安全禁发列表、数值物理范围、CRC与计数器规则、DLC长度以及报文发送速率上限。该模块会输出详细的拦截统计报告，记录被拦截条数及主要原因分布。
- 注入与监测模块： 负责与测试平台的物理接口。它将通过约束器的测试序列按既定节奏注入到CAN总线，并同步、高频率地采集VCU的响应报文、内部状态标志、总线错误帧以及系统负载等实时数据，为后续判定提供原始依据。

- 判定与回灌模块： 实现测试闭环的智能反馈。它将监测到的系统实际行为与“答案卡”（即基于JSON配置的期望结果模板）进行比对，将结果判定为“正常”、“新状态”或“疑似Bug”。对于成功触发异常的测试用例，其上下文信息（如信号组合、时序特征）会被打标，并在后续的GAN采样中被赋予更高权重，引导模型向更高效的测试区域探索。
- 报告渲染模块： 负责所有测试信息的聚合与可视化输出。它整合各模块的运行数据，生成包含模型版本对比、测试覆盖度、异常多样性、TopN用例详情与拦截统计的综合报告，并支持一键导出为PDF或Markdown格式。
- Web前端： 为测试工程师提供统一的图形化操作界面。其核心功能包括：测试计划创建向导、任务执行实时监控（动态数据曲线与事件日志）、异常案例详情查看、测试报告中心以及模型与数据版本的只读管理页面。

5. 解决方案路径（Solution Approach）

5.1 系统整体结构

整个系统通过“数据库中心化 + 双模输入生成 + 统一执行接口 + 反馈闭环”的机制，实现了从传统规则模糊测试到智能 GAN 模糊测试的平滑过渡，使输入生成更加智能，测试覆盖更全面，异常发现更高效。



系统由五个核心模块组成：

a. 北汽被测平台

- 实际运行 VCU 模块（或其他控制器）的测试平台（HIL 台架）。
- 负责接收测试输入信号，执行对应功能（如休眠/唤醒），并返回系统响应、状态、日志等数据。

b. 用例执行模块

系统的“中枢控制器”，负责：

- 从数据库读取测试输入数据；

- 向被测平台发送变异因子（即测试输入信号）；
- 收集被测平台的输出与运行状态；
- 将结果数据返回数据库或判定模块；
- 在不同模糊测试模式之间切换（常规 or GAN）。

c. 数据库

- 储存所有输入样本、测试参数、期望输出、执行日志和模型信息。
- 是 常规模糊测试因子变异 和 GAN 模糊测试因子变异 的统一数据源。
- 同时记录执行结果供后续分析与模型训练。

d. 常规模糊测试因子变异模块

- 基于规则的传统变异模块。
- 取出数据库中的样本数据，按预设策略生成变异输入：
  - 单参数变异、多参数组合、时序扰动、重复执行等；
  - 输出“变异后的输入信号序列”。
- 将生成的变异因子发送至“用例执行模块”。

e. GAN 模糊测试因子变异模块

- 基于 GAN 模型的智能变异模块。
- 从数据库提取原始样本、特征范围、历史异常数据，用于模型采样。
- GAN 生成看似“真实但潜在异常”的输入数据，再通过约束器过滤。
- 最终将合规的变异因子发送至“用例执行模块”。

5.2 系统运行机制

整个系统运行流程可以分为数据准备、输入生成、测试执行、结果回传四个主要阶段。数据从数据库取出 → 经过常规或 GAN 变异生成输入 → 由执行模块发送到被测平台 → 收集结果回写数据库 → 异常样本回馈形成测试闭环。具体如下：

a. 数据准备

- 数据库存储正常样本、异常样本、参数范围、规则等信息。
- 用例执行模块根据任务配置，从数据库中读取输入数据。

b. 输入生成（两种模式）

- 常规模糊测试变异模块：按固定规则（单参数、多参数、时序扰动等）生成测试输入。
- GAN 模糊测试变异模块：基于历史数据由 GAN 模型生成“接近异常”的智能输入。
- 两者生成的数据格式一致，均可直接送往执行模块。

c. 用例执行与平台交互

- 用例执行模块将变异后的输入因子发送到北汽被测平台。
- 平台执行休眠/唤醒等功能，返回状态、故障码、信号响应等结果。

d. 结果记录与反馈

- 执行模块收集平台返回的数据并写入数据库。
- 若发现异常或新状态，生成新的测试用例或训练数据，供下一轮测试或模型优化使用。

5.3 将使用的平台、工具、库等资源

后端：Python、FastAPI（REST/WS）、PyTorch（GAN）、Pandas、Jinja2/WeasyPrint（报告）、Docker。

前端：React/Next.js（或 Vite + React）、ECharts/Chart.js、WebSocket、JWT、Ant Design。

数据/接口：DBC解析工具、BLF/ASC/CSV 读写、（可选）SocketCAN/CAN 网关 SDK。

5.4 测试方案

测试分为离线准备 → 在线注入 → 判定与反馈 → 复现与归因四阶段，结合具体策略（0–3）实现自适应测试。具体如下：

a. **离线准备**

· JSON配置：明确每个信号的type、range、step、单位、约束、是否作为触发点、预期输出模板（正常/异常的标志字段）。

· 训练数据：收集正常运行与历史异常样本（BLF/ASC/CSV），用于GAN训练及统计指标（分布、时序特征）。

· GAN训练：训练条件GAN（条件可为模式位/使能位等），并保留判别器/生成器性能指标（KL/JS、FID/其他距离度量）。

b. **输入生成与注入策略（变异模块细化）**

· 策略0（无约束变异）：按均匀分布或历史分布抽样初始输入列表，作为探索基线；列表中元素被取出后清除（防止重复抽取）。

· 策略1（单参数逼近）：当单参数触发新状态或异常时，将触发值附近若干值插入队列最前端（越有问题越靠近），提高局部采样频率以逼近临界边界。

· 策略2（多参数局部组合）：当多参数组合触发新状态时，针对已变异参数增加若干局部组合样本并前置到队列中，扩大局部组合搜索。

· 策略3（重复执行）：针对执行时间长或怀疑为资源问题的组合，向队列前端插入 >=20 条重复用例以增加重复验证机会（检测卡死/不稳定复现）。

· GAN 辅助生成：在策略0/1/2 中以一定比例替换或补充样本，生成“近似但不在培训集内”的边界样本，提高发现隐性缺陷能力。

· 正交设计：对于高维参数空间，使用正交表方法生成初始组合以保证覆盖均匀性并减少样本量。

c. **在线注入（HIL / 虚拟）**

· 场景构建：单域/多域并发、上/下电循环、唤醒/休眠链路干扰、网络拥塞/丢包/延迟、硬件故障模拟。

· 速率与配额：按实验计划设定注入速率（pkt/s）与每策略配额；支持A/B比较同配额下不同model_ver/temp/cond的效果。

· 复位与隔离：每轮或按策略在注入前进行系统复位以保证初始状态一致，或在需要时保留状态以测试长期累积效应。

d. **判定与反馈**

· 实时判定：Evaluator定时拉取平台状态并与JSON预期对比，若检测到新状态或异常返回策略信号（0–3）并写入异常指纹库。

· 趋势分析：检测十次内线性增长的响应时间（怀疑内存泄漏）、单组合执行时间异常增大（怀疑阻塞/死锁）、功耗不回落（怀疑唤醒/休眠逻辑问题）。

· 自动化回放：对高危异常自动触发回放流程，执行最小化搜索以生成最小触发用例，便于开发定位。

· 报告：每轮输出包含：用例列表、异常指纹、复现率、覆盖度更新、策略调整记录的报告（HTML/PDF）。

5.5 测试策略评估方法

测试策略的充分性通过以下四类指标综合评估：

a. **覆盖度指标**

· ID/字段/时序覆盖率：统计被测信号、参数取值和关键时序（如唤醒Δt、保持时间）被测试的比例。

· 当新增覆盖率持续下降并趋于稳定时，说明测试空间已基本探索完毕。

b. **异常发现与收敛性**

· 绘制“去重后异常指纹数量—测试轮数”曲线：若新增异常指纹趋近于零，表明策略已趋于收敛；若仍增长明显，则需调整或扩展策略。

c. **复现率与稳定性**

· 统计高危异常的复现成功率。复现率高说明策略稳定可靠；低复现率则提示需要优化输入建模或环境条件。

d. **边界与显著性验证**

· 对关键边界参数进行细粒度扫描，验证临界点两侧均被覆盖。

· 比较不同策略或模型版本在相同条件下的异常发现率，若差异显著，则更新主策略。

6. 项目管理（Project Management）

6.1 开发流程

我们计划采用敏捷开发模式，以两周为一个迭代周期。每个周期都致力于交付一个可验证、可演示的软件增量。团队每周将通过会议同步进展，并利用迭代评审会来收集反馈和调整后续计划，确保项目能灵活应对挑战并始终朝着核心目标推进。

6.2 最小可行系统

在首个迭代周期中，我们计划构建的最小可行系统将聚焦于核心流程的打通。该系统将包含一个基础的条件GAN模型，用于生成受限的测试信号；一个严格的统一约束器，用于确保生成信号的基本安全性与合规性；一个虚拟注入器，用于模拟与VCU控制器的交互；以及一个最小化的Web控制界面，支持创建测试任务、监控关键状态并查看生成的测试报告。此MVP旨在快速验证从“智能生成”到“注入反馈”的闭环可行性。

6.3 愿望清单

在最小可行系统之上，我们的功能愿望清单包括：实现能够更精细捕捉时序特征的循环生成对抗网络；开发异常模式的自动聚类与归因分析功能，以提升缺陷定位效率；以及构建一个交互式的测试序列可视化回放模块，用于深度分析异常触发的全过程。

6.4 团队协作

团队工作将围绕模块分工进行协调。林琪同学负责数据基础与约束层，张诗蔻同学攻坚GAN生成引擎，达思睿同学负责系统集成与报告输出。胡宝怡同学统筹规划项目，负责协调各方进度及主持会议，同时负责前端开发，并视情况协助或主导后端核心模块的实现，确保系统架构前后统一。

我们计划每两周举行一次核心同步会议，聚焦于任务规划与技术方案对齐，代码审查、迭代演示与复盘，确保团队步调一致并高效解决开发中遇到的问题。

7. 团队（Team）

7.1 背景

我们团队的四名成员均具备机器学习、后端开发和Web前端的基础知识，这为项目的技术实现提供了基本保障。团队成员曾参与过软件开发相关的课程项目，但尚未有构建与此项目完全相同的工业级模糊测试系统的经验。对于汽车领域的硬件在环测试具体实践和生成对抗网络的深度应用，这对我们而言是全新的技术挑战，相关的工具链（如PyTorch for GAN, HIL测试设备操作）也需要在项目初期集中学习。

7.2 角色

基于团队成员的技术兴趣和项目需求，我们设定了初步的角色分工：林琪同学负责数据与约束模块，确保输入数据的规范性和生成信号的安全性；张诗蔻同学负责GAN核心引擎，致力于生成更有效的测试用例；达思睿同学作为系统集成与输出核心，负责连接测试环境、判定结果并生成最终报告。胡宝怡同学为该项目组长，将统筹项目的整体技术规划、进度管理与架构设计，并主导前端开发，并视情况协助或主导后端核心模块的实现。

8. 约束与风险（Constraints and Risks）

8.1 约束

我们计划采用敏捷开发模式，以两周为一个迭代周期。每个周期都包含明确的需求定义、方案设计、代码实现、测试评估和复盘总结环节。为确保沟通顺畅，团队每周将进行一次约30分钟的会议，同步进度并解决阻塞问题，同时统一项目中涉及的术语和关键指标的定义，避免后续理解偏差。

8.2 资源与风险

在资源与依赖方面，项目的成功高度依赖于能否从合作企业方及时获取准确的系统设计文件（如DBC文件、报文规范）、真实的正常通信日志、信号白名单与禁发列表等关键信息。若这些资源无法及时到位，我们将启动备选方案，使用虚拟注入器进行系统核心功能的开发与演示，并在所有产出报告中明确标注所使用的数据口径。

在技术实现上，主要风险在于GAN模型训练不稳定或生成的有效测试用例比例过低。为此，我们拟定了多级应对策略：首先从结构简单的小模型开始，配合早停和正则化技术；其次，实施失败重采样机制，并对成功触发异常的测试上下文进行“提权”，从而引导模型向更有效的方向演化。最后，在项目进度方面，为了确保项目核心价值得以尽早验证，我们将优先集中资源交付一个基于虚拟注入器的、功能完整的可演示闭环。这样即使HIL测试接口的开放有所延迟，也不会影响项目主线的推进，待接口就绪后仅需替换连接层即可快速对接。