Engineering Trustworthy Intelligent Systems

ETIS COMP330 Flagship Implementation Guide

Loyola University Chicago Reference Implementation

Software Engineering, Governance, and Operational Trust in the AI Era

William T. O'Connell, Ph.D.

FIRST EDITION EDUCATIONAL PRODUCT

Contents

Copyright and Use

This educational product is part of the Engineering Trustworthy Intelligent Systems educational product suite.

The ETIS book, appendices, website content, downloadable materials, figures, educational products, and related publication assets are protected by applicable copyright laws.

Educators and institutions are encouraged to use ETIS concepts, practices, and educational models in their own courses and programs. Redistribution or reproduction of substantial ETIS content, educational products, figures, appendices, downloadable materials, or instructor resources remains subject to the applicable ETIS licensing terms unless separate permission has been granted.

ETIS educational products are intended for professional and educational use. They are not substitutes for institutional policy, legal advice, regulatory review, academic governance, or professional judgment.

ETIS Educational Product Series

This document is part of the **ETIS Educational Product Series**.

The educational product series transforms *Engineering Trustworthy Intelligent Systems* from a publication into a teachable, adoptable, and stewardable engineering framework.

The series is designed for instructors, students, departments, universities, professional educators, and institutions adopting ETIS in software engineering, AI governance, responsible AI, enterprise systems, capstone, project-based, or professional-practice environments.

Product Family

The ETIS Educational Product Series includes:

Product	Primary Purpose
ETIS Educational Ecosystem Guide	Explain the full ETIS educational architecture and public product model
ETIS Instructor Course Package	Provide the instructor operating system for course design and delivery
ETIS Classroom Facilitation Guide	Help instructors run ETIS classrooms as active engineering environments
ETIS Instructor Handbook	Preserve instructor guidance, teaching judgment, and long-term stewardship practices
ETIS Student Professional Engineering Guide	Help students understand and practice professional engineering behaviors
ETIS COMP330 Flagship Implementation Guide	Document the Loyola COMP330 reference implementation as a working educational laboratory

Relationship to the ETIS Book

The ETIS book remains the authoritative doctrine.

The educational product series translates that doctrine into teaching, learning, adoption, classroom operation, implementation, and stewardship resources.

Educational products teach ETIS.

Adoption examples prove ETIS.

Educational stewardship sustains ETIS over time.

Common Educational Premise

ETIS education is built on a simple premise:

> AI can produce artifacts. Engineers create trust.

Students should not merely complete assignments.

They should develop evidence of engineering maturity.

Instructors should not merely deliver content.

They should operate educational systems that help students practice trustworthy engineering.

Shared Educational Mission

ETIS educational products teach future engineers to:

- define intent
- engineer context
- bound authority
- verify behavior
- operate reality
- explain decisions
- own outcomes

ETIS COMP330 Flagship Implementation Guide

Who This Is For

This guide is for:

- instructors adopting ETIS in software engineering courses
- faculty evaluating ETIS for university use
- departments considering ETIS-based curriculum integration
- instructors teaching project-based software engineering
- instructors teaching AI-era software engineering practice
- capstone coordinators
- graduate teaching assistants
- educational leaders evaluating evidence-centered engineering education
- professional educators adapting ETIS for workforce development

It is especially useful for instructors who want to see how ETIS can operate in a real course rather than remain an abstract framework.

Purpose

The **ETIS COMP330 Flagship Implementation Guide** documents the Loyola University Chicago COMP330 reference implementation as a practical educational laboratory.

Its purpose is not to present COMP330 as the only correct way to teach ETIS.

Its purpose is to show how ETIS doctrine can be translated into a live software engineering course with real students, real teams, real constraints, real assignments, real repository evidence, real AI-use questions, and real instructional tradeoffs.

This guide helps educators understand:

- how ETIS can be implemented in a semester course
- how a course can become an educational proof environment
- how students can move from assignment completion toward engineering accountability
- how AI use can be normalized without surrendering engineering responsibility
- how repository-centered engineering can become the backbone of the course
- how release defense and operational thinking can reshape software engineering education
- how course evidence can drive future improvement

Relationship to ETIS

ETIS is the framework.

COMP330 is a reference implementation.

The relationship is:

ETIS Doctrine ↓ Educational Ecosystem ↓ Instructor Course Package ↓ COMP330 Flagship Implementation ↓ Educational Evidence and Future Evolution

The ETIS book remains authoritative.

The COMP330 implementation demonstrates one way to operationalize ETIS in an undergraduate software engineering context.

Institutions should inherit the principles, not copy every local detail.

Relationship to COMP330

COMP330 is treated as the flagship ETIS implementation because it provides a concrete educational environment for applying the framework.

The course demonstrates how ETIS can shape:

- course purpose
- student expectations
- repository structure
- team practices
- AI-use policy
- assignment sequencing
- milestone reviews
- project cycles
- release readiness
- assessment
- classroom facilitation
- instructor reflection
- continuous improvement

COMP330 is not merely a course using ETIS vocabulary.

It is an ETIS educational laboratory.

What Makes COMP330 a Flagship Implementation

The COMP330 implementation matters because it connects ETIS to a real teaching environment.

It provides evidence for questions such as:

- Can students understand evidence-centered engineering?
- Can students use AI responsibly when expectations are explicit?
- Can teams learn to treat repositories as engineering memory?
- Can release defense change student behavior?
- Can operational thinking be introduced before professional practice?
- Can ETIS scale into a real university course?
- Can course artifacts become reusable educational assets?
- Can instructor observations improve the framework over time?

The flagship implementation exists to answer those questions through practice.

Implementation Boundary

This guide is intentionally implementation-specific.

It may include Loyola-specific and COMP330-specific material, including:

- course context
- semester observations
- implementation timeline
- repository patterns
- student patterns
- adoption insights
- adaptation notes
- future evolution paths

Those details are valuable because they show how ETIS behaves in the real world.

However, they should not be mistaken for universal requirements.

How To Use This Guide

Use this guide after reading or reviewing the broader ETIS educational products.

Recommended use:

1. Read the **ETIS Educational Ecosystem Guide** to understand the overall educational architecture.
2. Read the **ETIS Instructor Course Package** to understand course design and instructional operating structure.
3. Read this guide to see how those ideas were implemented in COMP330.
4. Use the **Classroom Facilitation Guide** and **Instructor Handbook** to support actual teaching.
5. Adapt the COMP330 lessons to your own course, institution, students, schedule, and constraints.

This guide should be read as a reference implementation, not a mandate.

What This Guide Is Not

This guide is not:

- a universal syllabus
- a required course template
- a Loyola-only artifact
- a replacement for the ETIS book
- a substitute for local academic policy
- a guarantee that every course should follow COMP330 exactly

It is a practical implementation record.

Its value is in showing what happens when ETIS becomes real.

Core Implementation Thesis

The core thesis of the COMP330 flagship implementation is:

> A software engineering course should not merely teach students how to build software. It should teach them how to produce evidence that their engineering work is understandable, reviewable, verifiable, governable, operable, and defensible.

That is the implementation center of gravity.

Educational Lessons Preserved

The COMP330 implementation helps preserve lessons about:

- student transformation
- course sequencing
- AI normalization
- team accountability
- repository-centered engineering
- milestone design
- review behavior
- instructor intervention

- release defense
- operational maturity
- semester evolution
- institutional adoption

These lessons should improve future ETIS courses and adoption models.

Relationship to Other ETIS Educational Products

This guide connects to the full educational product family.

Product	Relationship
ETIS Educational Ecosystem Guide	Provides the full educational architecture into which COMP330 fits
ETIS Instructor Course Package	Provides the reusable instructor course design model
ETIS Classroom Facilitation Guide	Supports the classroom practices visible in the COMP330 implementation
ETIS Instructor Handbook	Preserves the teaching judgment needed to operate and improve implementations
ETIS Student Professional Engineering Guide	Helps students understand the professional behaviors expected in COMP330-like courses

Bottom Line

The COMP330 flagship implementation is not important because it is the only way to teach ETIS.

It is important because it proves that ETIS can move from framework to classroom, from doctrine to student behavior, from educational theory to repository evidence, and from assignments to professional engineering accountability.

It is the working educational laboratory for ETIS.

Part I

Flagship Context

Implementation Overview

Purpose

This document explains how Loyola University Chicago COMP330/474 operationalizes the ETIS (Engineering Trustworthy Intelligent Systems) Educational Ecosystem.

The purpose of this implementation is to transform traditional software engineering education into a trustworthy engineering experience that more closely resembles professional engineering practice.

This implementation intentionally combines educational theory, professional engineering experience, repository-centered engineering, evidence-centered engineering, and responsible AI collaboration into a unified educational system.

The implementation is designed to create future engineers who can build systems that are understandable, reviewable, governable, operational, and trustworthy over time.

Core Philosophy

Educational work should resemble professional engineering work.

Students should not experience software engineering as a disconnected sequence of assignments.

Students should experience software engineering as a complete engineering system.

The implementation intentionally moves students away from:

> Complete a task.

Toward:

> Build systems that can be defended over time.

The educational objective is engineering accountability rather than assignment completion.

Bi-Directional Educational Laboratory

COMP330/474 is intentionally designed as a bi-directional educational laboratory.

ETIS informs the course.

The course continuously informs ETIS.

This feedback loop allows educational theory to remain grounded in educational reality.

The relationship is intentional and ongoing.

ETIS
↓
COMP330/474
↓
Educational Evidence
↓
ETIS Refinement
↓
Future Implementations

Educational systems become stronger when theory and practice continuously inform one another.

Educational Environment

The course operates as a semester-long engineering environment rather than a traditional lecture course.

Typical characteristics include:

Characteristic	Typical Implementation
Semester Length	15 weeks
Enrollment	Approximately 30 students
Audience	Undergraduate and graduate students
Team Size	5–6 students
Project Model	One project per team
Engineering Model	Two-Cycle Engineering Model
Repository Model	Repository-centered engineering
AI Philosophy	AI encouraged and governed

The objective is to create an authentic engineering experience.

Audience Model

The implementation intentionally combines undergraduate and graduate students.

COMP330 Junior and senior undergraduate students.

COMP474 Graduate students.

Graduate students are intentionally distributed across teams.

Graduate students frequently assume leadership roles and help elevate engineering expectations throughout the classroom.

Mixed teams create richer engineering experiences.

Engineering Lifecycle

The course operationalizes ETIS through a structured engineering lifecycle.

Intent

↓

Requirements

↓

Architecture

↓

Implementation

↓

Verification

↓

Release Readiness

↓

Operational Thinking

The lifecycle is iterative rather than strictly sequential.

Students continuously revisit earlier decisions as the project evolves.

Two-Cycle Engineering Model

The semester uses the ETIS Two-Cycle Engineering Model.

Cycle 1 Question:

Can it work?

Students focus on:

- understanding the problem
- requirements
- architecture
- early implementation
- initial validation

Cycle 2 Question:

Can it survive?

Students focus on:

- hardening decisions
- improving quality
- operational thinking
- release readiness
- long-term accountability

This progression helps students mature beyond implementation thinking.

Six Engineering Phase Gates

Engineering accountability is distributed throughout the semester.

Students complete six engineering phase gates.

The phase gates create visible checkpoints for engineering maturity.

The objective is continuous progress rather than end-of-semester compression.

Phase gates help:

- identify risks early
- reduce procrastination
- reinforce engineering discipline
- increase accountability
- create educational evidence

The phase gates operationalize ETIS throughout the semester.

Repository-Centered Engineering

Repository-centered engineering is foundational.

Students learn that repositories are engineering memory systems.

Repositories preserve:

- intent
- context
- decisions
- risks
- AI usage
- reviews
- testing evidence
- release evidence
- operational evidence

Repositories become engineering systems rather than code storage locations.

This creates continuity between educational and professional environments.

Professional Portfolio Philosophy

Students are not building class projects.

Students are building the early stages of professional portfolios.

Students are encouraged to select projects that may continue for one to two years beyond the course whenever practical.

The objective is to graduate students with evidence of engineering ability.

Students should leave the course with artifacts they can discuss during interviews and professional opportunities.

AI Operationalization

AI is encouraged throughout the entire engineering lifecycle.

Students are encouraged to use AI for:

- brainstorming
- requirements development
- architecture discussions
- implementation
- reviews
- testing
- operations
- reflection

However, AI remains bounded by engineering accountability.

The governing principle remains:

> AI proposes; engineers verify.

The course intentionally teaches responsible AI collaboration rather than AI avoidance.

Educational Outcomes

The implementation seeks to transform students into future trustworthy engineers.

Students learn to:

- define intent
- engineer context
- bound authority
- verify behavior
- operate reality
- explain decisions
- own outcomes

These outcomes extend beyond traditional software engineering education.

Relationship to Professional Practice

This implementation is heavily informed by approximately four decades of software engineering experience.

The implementation translates professional engineering methods into educational experiences.

The objective is not to simulate industry.

The objective is to normalize professional engineering behaviors early.

Students repeatedly experience engineering activities that mirror professional practice.

Relationship to Future Adopters

This implementation is intentionally transparent.

Future adopters should be able to understand:

- how ETIS was operationalized
- why implementation decisions were made
- what educational challenges emerged
- what improvements should occur over time

The implementation itself becomes educational evidence.

Core Thesis

COMP330/474 operationalizes ETIS by transforming a traditional software engineering course into a repository-centered, evidence-centered, AI-enabled engineering environment that continuously develops trustworthy engineering behaviors.

Implementation Timeline

Purpose

This document describes the high-level operational flow used to implement ETIS within Loyola University Chicago COMP330/474.

The objective is not to prescribe a specific weekly schedule.

The objective is to explain how the semester intentionally matures students from traditional software development thinking toward trustworthy engineering thinking.

This implementation uses progressive engineering maturity rather than isolated educational activities.

Students continuously build upon prior engineering work throughout the semester.

The semester intentionally resembles professional engineering environments.

Core Philosophy

Engineering maturity should develop progressively.

Students should not experience the semester as disconnected assignments.

Students should experience a continuous engineering journey.

Engineering accountability should increase throughout the semester.

The implementation intentionally moves students through the following progression.

Student

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

The timeline is designed to support this transformation.

Semester Structure

The implementation typically follows a 15-week semester.

The semester is organized around the ETIS Two-Cycle Engineering Model.

Course Foundation

↓

Cycle 1

↓

Cycle Transition

↓

Cycle 2

↓

Release Readiness

↓

Reflection and Future Growth

The timeline intentionally distributes engineering accountability across the entire semester.

Phase 1: Course Foundation

Students establish their engineering environment.

Activities typically include:

- forming teams
- understanding repository-centered engineering
- understanding ETIS principles
- selecting or refining projects
- establishing team expectations
- introducing AI governance

The goal is establishing engineering intent.

Students begin building engineering environments rather than simply starting projects.

Phase 2: Cycle 1 — Can It Work?

Cycle 1 focuses on engineering feasibility.

Students begin transforming ideas into engineering systems.

Activities typically include:

- defining requirements
- developing architecture
- initial implementation
- early validation
- establishing engineering evidence

Students begin learning that engineering work extends beyond implementation.

Cycle 1 answers the question:

> Can it work?

Phase 3: Cycle Transition

Cycle transition is intentionally important.

Students pause and evaluate progress.

Activities typically include:

- engineering reviews
- peer evaluations
- identifying risks
- identifying gaps
- strengthening accountability

Students learn that engineering work should be challenged before continuing.

The objective is improving engineering maturity before entering Cycle 2.

Phase 4: Cycle 2 — Can It Survive?

Cycle 2 shifts toward long-term thinking.

Students strengthen their systems.

Activities typically include:

- improving quality
- strengthening evidence
- improving reviewability
- operational thinking
- release preparation

Cycle 2 answers the question:

> Can it survive?

Students begin thinking beyond implementation.

Phase 5: Release Readiness

Students prepare to defend engineering decisions.

Activities typically include:

- evidence consolidation
- release preparation
- known limitation identification
- engineering demonstrations
- final accountability reviews

Students learn that:

> A demo is not operational proof.

Engineering claims should be defensible.

Phase 6: Reflection and Future Growth

The semester intentionally extends beyond course completion.

Students are encouraged to think about future development opportunities.

Activities may include:

- identifying future improvements
- planning long-term project continuation

- strengthening professional portfolios
- preparing interview narratives

Students are encouraged to continue projects beyond the semester whenever practical.

The objective is long-term engineering growth.

Six Engineering Phase Gates

Engineering accountability is continuously reinforced through six engineering phase gates.

The phase gates create visible checkpoints throughout the semester.

The objective is continuous progress.

The phase gates help:

- reduce procrastination
- identify risks earlier
- strengthen accountability
- reinforce engineering discipline
- create educational evidence

Engineering accountability becomes a semester-long activity rather than an end-of-semester activity.

AI Timeline Integration

AI is integrated throughout the entire engineering lifecycle.

Students are encouraged to use AI during:

- brainstorming
- requirements development
- architecture discussions
- implementation
- testing
- reviews
- release preparation
- reflection

AI usage remains governed and accountable throughout the semester.

The governing principle remains:

> AI proposes; engineers verify.

AI is intentionally normalized rather than isolated.

Graduate Student Leadership Integration

Graduate students are intentionally distributed throughout teams.

Leadership responsibilities naturally emerge throughout the semester.

Graduate students frequently help:

- guide discussions
- elevate expectations
- mentor undergraduate students
- strengthen engineering rigor

The objective is collaborative engineering growth.

Educational Feedback Loop

The semester itself continuously generates educational evidence.

ETIS

↓

COMP330/474

↓

Educational Evidence

↓

ETIS Refinement

↓

Future Implementations

The implementation intentionally contributes back to ETIS.

Educational theory and educational practice continuously strengthen one another.

Core Thesis

The implementation timeline intentionally transforms a 15-week semester into a continuous engineering maturity journey where students progressively learn to think, build, review, defend, operate, and improve engineering systems over time.

Implementation Adaptations

Purpose

This document explains how the ETIS (Engineering Trustworthy Intelligent Systems) Educational Ecosystem was intentionally adapted to Loyola University Chicago COMP330/474.

ETIS is designed to be broadly applicable across institutions.

Individual implementations should adapt ETIS to their own educational environments while preserving its core engineering philosophy.

This document records the intentional adaptations that allow ETIS to operate effectively within Loyola University Chicago.

These adaptations should not be viewed as universal requirements.

They are implementation decisions.

Future institutions should adapt ETIS to their own environments while preserving engineering accountability.

Core Philosophy

Institutions should inherit ETIS doctrine, not ETIS implementations.

The objective is not educational uniformity.

The objective is educational consistency.

Educational environments will differ.

Engineering principles should endure.

Institutional Context

Characteristic	Loyola Implementation
Institution	Loyola University Chicago
Course Numbers	COMP330 and COMP474
Discipline	Software Engineering
Semester Length	15 weeks
Student Population	Undergraduate and graduate students
Typical Enrollment	Approximately 30 students
Team Size	5–6 students
Project Model	One project per team
Engineering Model	ETIS Two-Cycle Engineering Model

These implementation decisions create the local educational environment.

Mixed Undergraduate and Graduate Environment

One of the largest adaptations is the intentional combination of undergraduate and graduate students.

COMP330 serves undergraduate students.

COMP474 serves graduate students.

Graduate students are intentionally distributed across teams.

Graduate students frequently assume leadership responsibilities.

The objective is collaborative engineering growth rather than educational separation.

Graduate students help elevate engineering maturity across the classroom.

Future institutions may choose different models.

The important principle is engineering mentorship.

Semester-Based Implementation

ETIS is not inherently tied to academic semesters.

COMP330/474 adapts ETIS into a 15-week semester structure.

Engineering accountability is intentionally distributed throughout the semester.

The implementation avoids concentrating accountability at the end of the course.

Future adopters may use:

- quarter systems
- trimester systems
- accelerated programs
- bootcamps
- corporate training programs

The timeline may change.

The engineering philosophy should remain intact.

Repository-Centered Engineering Adaptation

Repository-centered engineering is foundational to this implementation.

GitHub is required.

GitHub was selected because:

- students are already familiar with it
- it is freely available
- it is professionally relevant
- it supports repository-centered engineering

GitHub itself is not ETIS.

GitHub is the implementation mechanism.

The engineering principles remain independent of tooling.

Future institutions may choose alternative platforms.

The repository-centered philosophy should remain intact.

Professional Portfolio Adaptation

Traditional software engineering courses often optimize for assignment completion.

This implementation intentionally optimizes for professional portfolio development.

Students are encouraged to select projects that may continue for one to two years beyond the course whenever practical.

Students should graduate with evidence of engineering ability.

Students should leave the course with projects they can discuss during:

- internships
- interviews
- job searches
- professional networking opportunities

The objective is long-term engineering growth.

Six Engineering Phase Gates

Traditional courses frequently use a small number of major assignments.

This implementation intentionally distributes accountability across six engineering phase gates.

Phase gates create continuous visibility into engineering maturity.

The objective is to:

- reduce procrastination
- identify risks earlier
- reinforce engineering accountability
- create educational evidence

The implementation intentionally mirrors professional engineering checkpoints.

AI Integration Adaptation

AI is intentionally encouraged throughout the entire engineering lifecycle.

Students are encouraged to use AI for:

- brainstorming
- requirements analysis
- architecture discussions
- implementation
- testing
- reviews
- release preparation
- reflection

This differs from traditional approaches that limit or prohibit AI usage.

The educational objective is responsible AI collaboration.

The governing principle remains:

> AI proposes; engineers verify.

The implementation intentionally teaches AI responsibility rather than AI avoidance.

Professional Practice Adaptation

The implementation is heavily informed by approximately four decades of software engineering experience.

Professional experiences have been intentionally translated into educational experiences.

This includes:

- enterprise software engineering
- organizational transformation
- quality engineering
- delivery excellence
- consulting across hundreds of organizations

Leadership experiences also inform the implementation.

This includes:

- IBM Distinguished Engineer
- IBM CTO for Quality and Delivery Excellence

The objective is not to simulate industry.

The objective is to normalize professional engineering behaviors early.

Educational Feedback System Adaptation

Most educational frameworks operate in a one-directional manner.

Framework

↓

Course

↓

Students

This implementation intentionally creates a bi-directional feedback loop.

ETIS

↓

COMP330/474

↓

Educational Evidence

↓

ETIS Refinement

↓

Future Implementations

Educational experiences continuously improve ETIS itself.

This adaptation is intentional.

It allows ETIS to remain grounded in reality.

What Should Future Institutions Adapt?

Future institutions should adapt:

- calendars
- team sizes
- project domains

- student populations
- tooling choices
- local constraints

Future institutions should preserve:

- engineering accountability
- repository-centered engineering
- evidence-centered engineering
- AI governance
- operational thinking
- trustworthy engineering principles

Adapt the environment.

Preserve the doctrine.

Core Thesis

Loyola University Chicago demonstrates that ETIS can be intentionally adapted to local educational environments without sacrificing its engineering philosophy.

Local implementations should evolve.

Engineering principles should endure.

Educational Laboratory Insights

Purpose

This document captures the educational insights discovered through the ongoing implementation of ETIS (Engineering Trustworthy Intelligent Systems) within Loyola University Chicago COMP330/474.

These insights were discovered through real classroom experiences.

They represent observations about students, engineering behaviors, AI adoption, repository-centered engineering, accountability systems, and long-term educational outcomes.

The purpose is not to document mistakes.

The purpose is to preserve educational engineering memory.

These insights continuously contribute back to ETIS itself.

Core Philosophy

Educational laboratories are where educational frameworks become trustworthy.

Educational theory becomes stronger when continuously challenged by educational reality.

This implementation intentionally creates a bi-directional feedback loop.

ETIS

↓

COMP330/474

↓

Educational Evidence

↓

ETIS Refinement

↓

Future Implementations

Educational experiences should continuously improve educational frameworks.

Insight 1: Educational Work Should Resemble Professional Engineering Work

One of the earliest discoveries was that students respond positively when educational activities resemble professional engineering activities.

Students become more engaged when they understand that they are practicing real engineering behaviors rather than completing isolated academic exercises.

Professional engineering environments should not be introduced after graduation.

They should be introduced during education.

This insight significantly influenced ETIS.

Insight 2: Repository-Centered Engineering Creates Continuity

Repository-centered engineering proved highly effective.

Students quickly learn that repositories preserve more than source code.

Repositories preserve engineering memory.

Students begin seeing repositories as systems that preserve:

- requirements
- plans
- architecture
- AI usage
- reviews
- testing evidence
- release evidence
- operational thinking

This creates continuity between education and professional practice.

This insight became foundational to ETIS.

Insight 3: Accountability Should Be Distributed

Students naturally procrastinate when accountability is concentrated at the end of the semester.

Distributed accountability produces stronger engineering outcomes.

The six engineering phase gates intentionally address this challenge.

Smaller checkpoints create:

- continuous progress
- earlier intervention opportunities
- better visibility
- stronger engineering discipline

Engineering accountability should be continuous rather than episodic.

This insight significantly influenced ETIS.

Insight 4: Teams Need Early Intervention

Many teams naturally experience participation imbalances.

Most semesters include at least one team member who is not fully engaged.

These situations should be identified early.

Problems rarely resolve themselves without intervention.

Early visibility creates better outcomes.

Repository evidence and phase gates help surface these issues sooner.

Engineering environments should expose collaboration risks before they become larger problems.

Insight 5: AI Responsibility Outperforms AI Avoidance

Students who responsibly integrate AI often move faster than students who avoid AI entirely.

This became one of the strongest educational observations.

Students initially hesitant to use AI often struggle to keep pace with teams that responsibly adopt it.

Students should not be taught to avoid AI.

Students should be taught to govern AI.

The educational objective is AI responsibility.

The governing principle remains:

> AI proposes; engineers verify.

This insight heavily influenced ETIS.

Insight 6: Students Need Permission To Use AI Broadly

Many students initially assume AI should only be used for implementation.

Students should instead be encouraged to use AI throughout the entire engineering lifecycle.

Students should use AI for:

- brainstorming
- requirements
- architecture
- implementation
- reviews
- testing
- operations
- reflection

Students quickly learn that AI is an engineering collaborator rather than a coding tool.

This insight significantly expanded ETIS.

Insight 7: Students Build Better Projects When They Build For Themselves

Students become more invested when projects have meaning beyond the semester.

Students are encouraged to think about projects as professional portfolio assets.

Students should build systems they may continue developing for one to two years after the course whenever practical.

This changes student motivation.

Students begin thinking about long-term ownership rather than short-term completion.

Insight 8: Graduate Students Elevate Engineering Maturity

Mixed undergraduate and graduate teams create stronger engineering environments.

Graduate students frequently elevate expectations naturally.

Graduate students often provide:

- leadership
- mentorship
- perspective
- stronger engineering rigor

The objective is collaborative maturity rather than hierarchy.

This implementation model has proven valuable.

Insight 9: Students Want Real Engineering Experiences

Students consistently respond positively when educational activities resemble industry practices.

Students often express appreciation for learning approaches that feel authentic.

Students want to understand:

- how engineering actually works
- how teams collaborate
- how decisions are made
- how engineering accountability functions

Educational authenticity matters.

This insight strongly influenced ETIS.

Insight 10: Educational Systems Need Feedback Loops

Educational systems should continuously evolve.

Educational systems should never become static.

Educational evidence should be continuously collected.

Observations should continuously refine educational frameworks.

The relationship should remain bi-directional.

Educational laboratories strengthen educational systems.

Emerging ETIS Educational Doctrines

Several educational doctrines emerged directly from this implementation.

Educational work should resemble professional engineering work.

Engineering accountability is the educational outcome, not the side effect.

Students should graduate with evidence of engineering ability rather than evidence of course completion.

Repository-centered engineering creates continuity between education and professional practice.

AI responsibility outperforms AI avoidance.

Educational laboratories are where educational frameworks become trustworthy.

Core Thesis

COMP330/474 demonstrates that educational systems become stronger when educational theory and educational reality continuously inform one another.

The framework informs the course.

The course informs the framework.

The learners transform.

Part II

Implementation Notes

Adoption Insights

Purpose

This document captures insights that may help future institutions adopt and operationalize ETIS (Engineering Trustworthy Intelligent Systems).

These insights emerged from repeatedly teaching and evolving Loyola University Chicago COMP330/474.

The objective is not to prescribe a single implementation model.

The objective is to help future adopters understand what works, what requires intentional effort, and what should remain foundational.

This document should continuously evolve over time.

Core Philosophy

Institutions should inherit ETIS doctrine, not ETIS implementations.

Educational environments will differ.

Engineering principles should endure.

Successful adoption requires adaptation rather than replication.

Insight 1: Start With Educational Philosophy

Do not begin with tools.

Do not begin with assignments.

Do not begin with documentation.

Begin with educational philosophy.

Students should understand why ETIS exists before learning how ETIS operates.

The educational objective should remain clear.

The goal is trustworthy engineers.

Everything else supports that outcome.

Insight 2: Educational Work Should Resemble Professional Engineering Work

Students consistently respond positively when educational activities resemble professional engineering activities.

Professional engineering environments should be introduced during education.

Students should repeatedly experience:

- engineering accountability
- engineering reviews
- engineering evidence
- engineering tradeoffs
- engineering ownership

Authenticity matters.

Insight 3: Repository-Centered Engineering Is Foundational

Repository-centered engineering should remain foundational.

Students should not experience repositories as code storage locations.

Students should experience repositories as engineering memory systems.

Repositories should preserve:

- requirements
- plans
- architecture
- AI usage
- reviews
- testing evidence
- release evidence
- operational thinking

Repository-centered engineering creates continuity between education and professional practice.

Insight 4: Distributed Accountability Produces Better Outcomes

Students naturally procrastinate when accountability is concentrated near the end of the semester.

Continuous accountability produces stronger engineering outcomes.

The six engineering phase gates intentionally distribute accountability throughout the semester.

Continuous visibility creates:

- stronger engagement
- earlier interventions
- improved engineering discipline

Engineering accountability should be continuous.

Insight 5: AI Should Be Encouraged, Not Avoided

AI should be intentionally integrated throughout the engineering lifecycle.

Students should be encouraged to use AI for:

- brainstorming
- requirements
- architecture
- implementation
- reviews
- testing
- operations
- reflection

The educational objective is responsible AI collaboration.

The governing principle remains:

> AI proposes; engineers verify.

Teach AI responsibility rather than AI avoidance.

Insight 6: Students Need Authentic Projects

Students become more invested when projects have meaning beyond the semester.

Professional portfolio development creates stronger outcomes.

Students should build systems they may continue developing after course completion whenever practical.

Students should graduate with evidence of engineering ability.

Insight 7: Educational Systems Need Feedback Loops

Educational systems should continuously evolve.

Educational evidence should be intentionally collected.

Educational observations should continuously improve the framework.

The relationship should remain bi-directional.

ETIS

↓

Educational Implementation

↓

Educational Evidence

↓

ETIS Refinement

↓

Future Implementations

Educational laboratories strengthen educational frameworks.

Insight 8: Preserve Tool Independence

Tooling should support engineering philosophy.

Tooling should not become engineering philosophy.

Technology will evolve.

Engineering behaviors should endure.

Future institutions may choose:

- GitHub
- GitLab
- Azure DevOps
- other platforms

The repository-centered philosophy should remain intact.

Insight 9: Small Institutional Adaptations Are Expected

Institutions should adapt:

- calendars
- enrollment sizes

- team sizes
- project domains
- local constraints
- tooling choices

Institutions should preserve:

- engineering accountability
- repository-centered engineering
- evidence-centered engineering
- AI governance
- operational thinking
- trustworthy engineering principles

Adapt the environment.

Preserve the doctrine.

Insight 10: Educational Systems Should Be Sustainable

Educational environments should remain maintainable.

Avoid unnecessary complexity.

Avoid creating artifacts that instructors will not maintain.

Prefer lightweight systems that can endure over time.

Sustainability should be considered a design requirement.

Emerging ETIS Adoption Doctrines

Several adoption doctrines emerged through this implementation.

Educational work should resemble professional engineering work.

Repository-centered engineering creates continuity between education and professional practice.

Engineering accountability is the educational outcome, not the side effect.

Students should graduate with evidence of engineering ability rather than evidence of course completion.

Institutions should inherit ETIS doctrine, not ETIS implementations.

Educational laboratories are where educational frameworks become trustworthy.

Core Thesis

Successful ETIS adoption occurs when institutions intentionally adapt their local educational environments while preserving trustworthy engineering principles.

Repository Patterns

Purpose

This document captures common repository-centered engineering patterns observed over repeated offerings of Loyola University Chicago COMP330/474.

The objective is to understand how students evolve their repositories throughout the semester.

Students often arrive with prior GitHub experience.

However, many students have limited experience using repositories as engineering systems.

This document captures the transition from code repositories to engineering memory systems.

These patterns should help future instructors reinforce repository-centered engineering behaviors.

Core Philosophy

Students do not naturally think repository-centered.

Students learn repository-centered engineering through repeated practice.

Strong repositories preserve engineering memory.

Repositories should evolve beyond code storage.

Repositories should become engineering systems.

Common Repository Patterns

Pattern 1: Students Initially Treat Repositories As Code Storage Many students begin the semester thinking repositories exist primarily to store source code.

Students often undervalue engineering evidence.

This is expected.

Over time, students begin understanding that repositories preserve engineering work.

Students mature when repositories begin preserving:

- requirements
- plans
- architecture
- AI usage
- reviews
- testing evidence
- release evidence
- operational thinking

This transition should be intentionally reinforced.

Pattern 2: Repository Organization Often Emerges Slowly Students frequently delay organizing repositories.

Early repositories often appear incomplete or inconsistent.

Organization typically improves once engineering accountability increases.

Students benefit from explicit expectations early in the semester.

Strong starter kits accelerate this maturity.

Pattern 3: Students Initially Separate Code From Engineering Work Many students unintentionally separate implementation from engineering activities.

Students may initially believe:

> Source code is the project.

Students gradually learn:

> The repository is the project.

This is one of the most important maturity transitions in the course.

Pattern 4: Repository Quality Reflects Team Maturity Repository quality often becomes an early indicator of engineering maturity.

Well-maintained repositories frequently correlate with:

- stronger collaboration
- stronger accountability
- stronger communication
- stronger engineering discipline

Repository quality often reveals engineering maturity earlier than implementation quality.

Pattern 5: Engineering Evidence Often Improves Gradually Students frequently begin with minimal engineering evidence.

Over time, evidence quality improves.

Students begin preserving:

- decisions
- assumptions
- risks
- reviews
- limitations
- engineering rationale

Students learn that evidence is part of engineering work.

Pattern 6: Teams Benefit From Continuous Repository Updates Teams that update repositories continuously often outperform teams that defer repository work.

Continuous updates create:

- better visibility
- earlier interventions
- stronger accountability
- easier reviews

Repository maintenance should become routine behavior.

Pattern 7: Students Gradually Learn Traceability Students initially struggle to connect engineering artifacts together.

Over time, students begin linking:

- requirements

- plans
- architecture
- testing
- reviews
- releases

This creates engineering continuity.

Traceability should be intentionally reinforced.

AI Repository Patterns

Pattern 8: Students Initially Isolate AI Usage Students often begin by using AI only during implementation.

As students mature, AI usage expands throughout the lifecycle.

Students begin using AI for:

- brainstorming
- requirements
- architecture
- implementation
- reviews
- testing
- operations

AI usage becomes more integrated over time.

Pattern 9: Visible AI Usage Creates Better Outcomes Teams that document and discuss AI usage often demonstrate stronger engineering maturity.

Visible AI usage creates:

- accountability
- transparency
- better verification

Undocumented AI usage increases risk.

AI should remain visible.

Repository Maturity Signals

Repository maturity often appears when students begin asking:

- Can another engineer understand this repository?
- Is engineering evidence visible?
- What assumptions are preserved?
- What decisions are documented?
- What evidence supports our claims?
- What risks remain?

These are strong indicators of engineering growth.

Observation Guidelines

Add patterns only when they:

- appear repeatedly
- influence engineering outcomes
- benefit future instructors
- strengthen repository-centered engineering

Avoid isolated incidents.

Capture recurring trends.

Engineering Standard

Strong repositories preserve engineering memory rather than simply storing source code.

Semester Observations

Purpose

This document captures recurring observations that emerge from repeatedly teaching and evolving Loyola University Chicago COMP330/474.

The objective is to preserve implementation memory.

This document should capture patterns rather than isolated events.

Over time, these observations may improve both the course and ETIS itself.

Keep observations concise.

Core Philosophy

Educational systems improve when observations are intentionally preserved.

Small observations often become important insights.

Capture patterns.

Avoid recording every individual event.

Observation Log

Use this section to record meaningful observations after each offering.

Offering	Observation	Impact	Future Action

Common Areas To Observe

Student Engagement Observe:

- participation levels
- team interactions
- ownership behaviors
- accountability patterns

Engineering Maturity Observe:

- engineering decision quality
- repository usage
- evidence quality
- operational thinking

AI Usage Observe:

- AI adoption patterns
- AI strengths
- AI risks
- AI verification behaviors

Team Dynamics Observe:

- leadership behaviors
- collaboration patterns
- communication challenges
- workload distribution

Educational Friction Observe:

- areas of confusion
- recurring bottlenecks
- concepts requiring additional reinforcement

Examples Of Useful Observations

Examples may include:

- students adopt repository-centered engineering quickly
- students initially underestimate documentation value
- teams benefit from earlier accountability checkpoints
- students become more comfortable using AI over time
- operational thinking often develops later than implementation thinking

Record patterns rather than individual anecdotes.

Observation Writing Guidelines

Strong observations should answer:

- What happened?
- Why did it happen?
- Is this recurring?
- Should future instructors know this?
- Should ETIS evolve because of this?

Avoid:

- naming individual students
- semester-specific administrative details
- isolated one-time events

Relationship To ETIS

This document contributes to the ETIS feedback loop.

Educational Experience

↓

Observation

↓

Educational Memory

↓

ETIS Refinement

↓

Future Implementations

Educational experiences should continuously strengthen educational frameworks.

Engineering Standard

Educational memory should accumulate over time rather than reset every semester.

Student Patterns

Purpose

This document captures common student behaviors and learning patterns observed over repeated offerings of Loyola University Chicago COMP330/474.

The objective is not to identify weaknesses.

The objective is to help instructors anticipate how students typically mature throughout the semester.

Students often exhibit predictable engineering growth patterns.

Recognizing these patterns earlier allows instructors to intervene more effectively.

This document should evolve over time.

Core Philosophy

Students are not unfinished engineers.

Students are developing engineers.

The objective is not to eliminate these patterns.

The objective is to intentionally guide students through them.

Strong educational systems anticipate predictable behaviors.

Common Student Patterns

Pattern 1: Students Initially Think Like Implementers Students often begin the semester focused primarily on coding.

Many students initially believe software engineering is synonymous with implementation.

Students often underestimate:

- requirements
- architecture
- reviews
- operational thinking
- engineering evidence

This is expected.

Students gradually mature beyond implementation-focused thinking.

Pattern 2: Students Initially Underestimate Repository Value Many students initially see repositories as code storage locations.

Over time, students begin understanding repositories as engineering memory systems.

This transition is significant.

Students begin preserving:

- requirements
- plans
- decisions
- AI usage
- reviews

- evidence

This maturity should be intentionally reinforced.

Pattern 3: Students Often Delay Work Early In Cycle 1 Students frequently underestimate the amount of engineering work required early in the semester.

Teams often delay activities until accountability checkpoints appear.

The six engineering phase gates intentionally reduce this behavior.

Continuous accountability improves outcomes.

Early intervention is valuable.

Pattern 4: Team Participation Imbalances Are Common Many teams experience participation imbalances.

Most semesters include at least one student who is less engaged.

These situations should be identified early.

Problems rarely resolve themselves without intervention.

Continuous visibility helps.

AI Adoption Patterns

Pattern 5: Students Are Often Initially Hesitant To Use AI Many students are unfamiliar with using AI beyond implementation.

Students may initially avoid AI because:

- they are uncertain how to use it
- they fear misuse
- they lack confidence

Intentional encouragement is important.

Students should be taught that AI is an engineering collaborator.

Pattern 6: Responsible AI Users Often Progress Faster Students who responsibly integrate AI often make progress more quickly.

AI frequently accelerates:

- brainstorming
- requirements development
- architecture discussions
- implementation
- reviews

Students should not avoid AI.

Students should govern AI.

Professional Growth Patterns

Pattern 7: Students Become More Engaged When Projects Matter Students invest more effort when projects have meaning beyond the semester.

Professional portfolio discussions often increase engagement.

Students begin thinking differently when they understand they are creating evidence of engineering ability.

Long-term ownership creates stronger outcomes.

Pattern 8: Students Value Authentic Engineering Experiences Students consistently respond positively when educational activities resemble professional practice.

Students often appreciate:

- realistic expectations
- structured reviews
- engineering accountability
- repository-centered engineering
- AI integration

Authenticity matters.

Emerging Maturity Signals

Students often demonstrate increased engineering maturity when they begin asking questions such as:

- How will another engineer understand this?
- What evidence supports this decision?
- What risks remain?
- How do we defend this release?
- What happens after deployment?
- How should AI be governed?

These questions should be encouraged.

Observation Guidelines

Add new patterns only when they:

- appear repeatedly
- influence engineering outcomes
- benefit future instructors
- improve ETIS understanding

Avoid documenting isolated incidents.

Focus on recurring trends.

Engineering Standard

Strong educational systems anticipate student growth patterns and intentionally design learning experiences around them.

Part III

Evolution

Future Evolution

Purpose

This document describes how the Loyola University Chicago COMP330/474 implementation may continue evolving over time.

The objective is not to create a fixed roadmap.

The objective is to preserve a continuous improvement mindset.

This implementation intentionally operates as a living educational system.

Educational systems should continuously evolve as technology, engineering practices, AI capabilities, and student needs change over time.

The implementation should remain stable in philosophy while remaining adaptable in practice.

Core Philosophy

Educational systems should evolve.

Educational doctrines should endure.

The goal is not educational stability.

The goal is educational resilience.

Future evolution should preserve engineering accountability while allowing educational environments to adapt over time.

Long-Term Vision

The long-term objective is to continuously strengthen the relationship between ETIS and real educational environments.

ETIS

↓

Educational Assets

↓

COMP330/474

↓

Educational Evidence

↓

ETIS Refinement

↓

Future Implementations

The implementation should remain a continuous educational feedback system.

Evolution Area 1: Strengthen Student Onboarding

Students enter the course with varying levels of software engineering maturity.

Future implementations should continue improving onboarding experiences.

Areas for continued investment include:

- repository-centered engineering
- ETIS principles
- AI accountability
- engineering evidence expectations
- trustworthy engineering concepts

Students should establish strong foundations early.

Early confusion often creates downstream friction.

Evolution Area 2: Strengthen AI Responsibility

AI capabilities will continue evolving rapidly.

Educational environments should evolve alongside them.

Future implementations should continue strengthening:

- AI governance
- AI verification
- AI disclosure
- AI risk awareness
- AI engineering workflows

Students should become responsible AI collaborators.

AI responsibility should become a normal engineering behavior.

Evolution Area 3: Strengthen Engineering Review Culture

Students often arrive with limited experience participating in formal engineering reviews.

Future implementations should continue strengthening:

- review board experiences
- peer reviews
- engineering defenses
- evidence evaluations
- engineering discussions

Engineering work should become increasingly defendable over time.

Evolution Area 4: Strengthen Repository Analysis

Students should become increasingly sophisticated in how they manage repositories.

Future implementations should continue strengthening:

- repository organization
- engineering evidence integration
- traceability
- repository hygiene
- repository storytelling

Repositories should continue evolving into engineering memory systems.

Evolution Area 5: Strengthen Professional Portfolio Development

Professional portfolio development should continue becoming more intentional.

Students should graduate with demonstrable engineering evidence.

Future implementations should continue helping students build artifacts they can use throughout their careers.

Students should be able to explain:

- what they built
- why they built it
- how they verified it
- what tradeoffs they made
- how AI was governed
- how risks were managed

Students should leave with engineering narratives rather than isolated assignments.

Evolution Area 6: Strengthen Operational Thinking

Operational thinking should continue moving earlier in the semester.

Students should begin thinking beyond implementation sooner.

Future implementations should continue strengthening:

- release readiness
- observability
- operations
- reliability thinking
- long-term ownership

Students should increasingly think like system stewards.

Evolution Area 7: Expand Future Institutional Adoption

This implementation should eventually serve as a model for future adopters.

Future implementations should help other institutions understand:

- what should be preserved
- what may be adapted
- how engineering accountability is maintained
- how ETIS is operationalized

The goal is educational scalability rather than educational replication.

Institutions should inherit doctrine, not implementations.

Evolution Area 8: Strengthen Educational Evidence Collection

Educational observations should continue being preserved.

Educational systems improve when evidence is continuously collected.

Future implementations should continue capturing:

- student patterns
- engineering behaviors
- AI usage patterns
- team dynamics
- repository patterns
- adoption insights

Educational memory should not be lost between semesters.

Evolution Area 9: Preserve Professional Relevance

Technology will evolve.

Tools will evolve.

AI will evolve.

Professional expectations will evolve.

The implementation should continuously adapt while preserving its engineering philosophy.

Students should continue learning durable engineering behaviors rather than transient technologies.

The implementation should remain future oriented.

What Should Never Change

Several principles should remain stable over time.

Educational work should resemble professional engineering work.

Repository-centered engineering should remain foundational.

Engineering accountability should remain continuous.

AI should remain governed and accountable.

Students should graduate with evidence of engineering ability.

Operational thinking should remain integral.

Educational systems should remain bi-directional.

The framework should inform the course.

The course should inform the framework.

Core Thesis

The future of COMP330/474 is not to become a static software engineering course.

Its future is to remain a continuously evolving educational system that transforms students into future trustworthy engineers while simultaneously strengthening ETIS itself.