FIT2001 Systems Development

FACULTY OF INFORMATION TECHNOLOGY · MONASH UNIVERSITY · SEMESTER 2, 2026

EXAM REVISION SHEET
Side 1 · rules + methods
Scheduled final · 40%

Full guide + ask Sia about FIT2001

SIDE 1/2 FINAL PRIORITIES · DEFINITIONS · DECISION RULES · MODELS · TRACEABILITY

LAST-WEEK REVISION

Two hours 10 minutes · 45% component threshold

01 · Exam map

COURSE · BRIEF

Tut 16 | A1 | A2 23 | A3 14 | Final 40%

COMPONENT	WEIGHT
Tutorial participation + pre-class activities	16%
Assignment 1 / 2 / 3	7% / 23% / 14%
Scheduled final assessment	**40%**

OPERATIONAL FACT

```
Final = 2 h 10 min · 45% component threshold.
```

Permitted materials: not confirmed for this offering in the course materials we hold — check the exam instructions on the LMS.

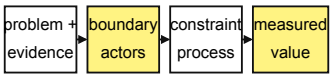
FINAL-ANSWER MOVE	WHAT EARNS THE LINE
Define	Give the term its distinct analytical job.
Apply	Use named actors, conditions, rules and evidence from the case.
Connect	Show what the artefact supplies to the next artefact.
Qualify	Test an alternate, misuse, failure or change condition.
Conclude	State the justified decision in case language.

PRIORITY *Earn marks by linking evidence → requirement/model → design choice → verification; lists without a changed conclusion are weak.*

02 · Problem + boundary

C1

System boundary = explicit division between what the proposed system controls and what remains environment or external actor. ***Stakeholder*** = person, group or organisation affected by or able to affect the system and outcomes. ***Systems development*** = coordinated analysis, design, implementation and change work solving an organisational problem through an information system.



Problem framing precedes technology choice.

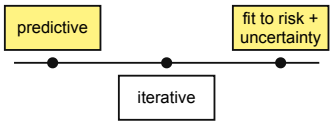
ASK	CHECKABLE ANSWER
Who/why?	Affected actors, goals, conflicts
Where?	Inside/outside boundary; dependencies
Evidence?	Current process, causes, constraints
Success?	Measures independent of one interface

BOUNDARY AUDIT

03 · Lifecycle choice

C2

Predictive lifecycle plans and baselines substantial scope before later construction and release. ***Iterative development*** uses repeated cycles to refine understanding and solution through feedback. ***Method fit*** aligns the approach with uncertainty, risk, governance, team and stakeholder conditions.



Method labels follow project conditions.

IF THE CASE STRESSES…	RESPONSE
Stable scope, fixed compliance/interface	Baseline and staged controls
Unclear workflow, feedback need	Iterate with representative users
Both	Hybrid: control fixed risks; iterate uncertainty

Compare consequences, not labels. Name what is baselined, what is iterated and which risk each choice controls.

EXAM COMPARISON

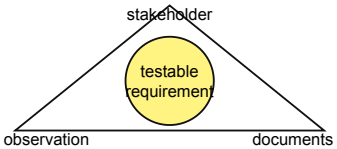
```
Conditions → commitment timing → feedback cadence → governance/control → justified fit.
```

TRAP *Calling a process agile does not remove governance, architecture or documentation needs.*

04 · Elicit evidence

C3

Elicitation purposefully discovers needs, rules, constraints and disagreement through stakeholder and artefact evidence. A ***functional requirement*** states capability or behaviour under stated conditions; a ***non-functional requirement*** states a quality, constraint or operating condition such as security, performance, accessibility or reliability.



Triangulate stakeholders, work and policy.

TECHNIQUE	BEST EVIDENCE / RISK
Interview	Goals, rationale / self-report
Observation	Actual steps, exceptions / observer effect
Documents + records	Rules, transactions / may be stale
Workshop	Conflict + agreement / power effects

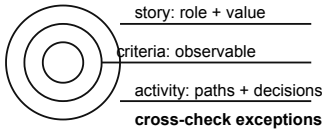
05 · Story ↔ activity

C4

User story connects role, desired capability and value; ***acceptance criteria*** make satisfactory behaviour observable; an ***activity diagram*** exposes actions, decisions, concurrency and process flow. Each checks a different gap.

STORY SCAFFOLD

```
As a [specific role], I want [capability], so that [decision/value].
Given [state], when [event], then [observable result].
```



Story + criteria + activity expose gaps.

DIAGRAM MARK	MEANING TO STATE
Action / arrow	Work / control flow
Decision + guards	Mutually meaningful branch conditions
Fork / join	Concurrency begins / synchronises
Start / end	Process scope and termination

THREE-WAY CONSISTENCY

STORY SAYS…	CRITERIA MUST…	ACTIVITY MUST…
Role + value	Observe value-producing behaviour	Show that role's path
Capability	Set success and boundary conditions	Show actions and decisions
Exception implied	Name acceptable result	Expose alternate/recovery path

Every criterion should map to an activity state, action or guarded outcome; every material branch should be covered or marked unresolved.

TRAP *A story card without conditions and alternate paths is not a complete specification.*

06 · Use-case contract

C5

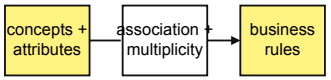
Use case = goal-oriented actor–system interactions under stated preconditions. The ***main success scenario*** achieves the goal normally; an ***alternate flow*** is a valid deviation, exception or failure tied to a main-flow step.

WRITE IN THIS ORDER	MARKER CHECKS
Actor, goal, trigger	Business goal, not screen task
Preconditions	True before interaction starts
Main steps	Actor intention ↔ system response
Alternates	Attach to exact numbered step

07 · Domain model + rules

C6

Domain model represents important real-world classes, attributes, associations and multiplicities. ***Multiplicity*** constrains instance counts at an association end. A ***business rule*** constrains or derives behaviour, structure or decisions independently of screen layout.



Business rules control the domain model.

QUESTION	MODEL MOVE
Thing with identity?	Candidate class
One value owned by one instance?	Attribute on that class
Relationship has status/date?	Association class
How many can relate?	Multiplicity at both ends
What must never/always occur?	Explicit rule/constraint

MODEL VALIDATION

Read each association both ways and test both end counts. A per-relationship value belongs on the relationship concept.

RULE FORM	MODEL CONSEQUENCE
"Each X must have…"	Minimum multiplicity or required relation
"At most one active…"	Maximum/uniqueness constraint
"Status depends on…"	Derived value or decision rule
"Only role R may…"	Behaviour/access constraint

OWNERSHIP RULE

```
Put an attribute on the concept whose one instance owns one value.
```

TRAP *Copying screens or tables can omit real business meaning.*

08 · Prototype to decide

C7

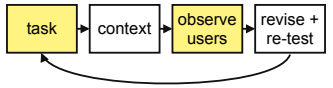
Prototype = deliberately incomplete representation used to learn about requirements, interaction or feasibility. ***Design constraint*** limits acceptable choices. ***Feedback loop*** converts observation into a revised requirement or design decision.

UNCERTAINTY	PROTOTYPE FOCUS
Language/process	Low fidelity; sequence + exceptions
Interaction	Task flow, feedback, recovery
Technical feasibility	Targeted executable spike
Before session	Name question, users, evidence, decision

09 · HCI judgement

C8

Usability asks whether specified users achieve goals effectively, efficiently and satisfactorily in context. ***Accessibility*** enables varied abilities and assistive technologies to perceive and operate. ***Interaction feedback*** communicates status, result, error and recovery.



Evaluate tasks with users in context.

IF YOU SEE…	DO / TEST
Invalid input	Specific nearby message; preserve valid work
Long operation	Status, progress, completion/failure
Keyboard/screen reader	Focus order, labels, announced errors
Destructive action	Prevent, confirm where needed, recover

State the user, task and context; the success, error or recovery evidence; keyboard/assistive paths; and the decision that changes if evidence fails. Name the observation, not merely the intended interface feature. Tie each observation to a revision.

ANSWER PATH

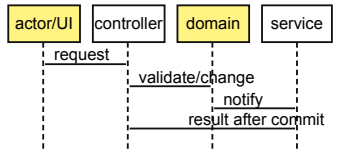
```
User + goal + context → observed breakdown → interface response → accessible evaluation.
```

TRAP *Familiar visual style does not prove usability or accessibility.*

10 · Realise the use case

C9

Use-case realisation describes collaborating objects/components performing a flow. A ***responsibility*** is a coherent obligation assigned without unrelated concerns. A ***sequence interaction*** orders messages to reason about control and data.



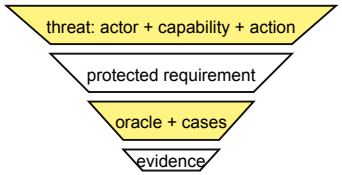
Messages expose ownership and failure order.

CHECK	WHY IT BITES
Start from preconditions + alternatives	Preserves requirements intent
Assign policy/state ownership	Avoids bloated controller

11 · Security + test evidence

C10

Threat scenario names an actor, capability and action that could violate a protected property. A ***test oracle*** defines the expected outcome for judging observed behaviour. A ***traceability matrix*** maps requirements, design elements and verification evidence.



Test misuse as well as normal success.

TRACE ROW	REQUIRED CONTENT
Need/risk	Stakeholder evidence or threat
Requirement	Observable protected behaviour
Design	Control/component responsibility
Verification	Case, oracle, result, residual risk

Coverage follows risk. State misuse actor/capability, control, adversarial case, oracle and residual risk.

ADVERSARIAL SET

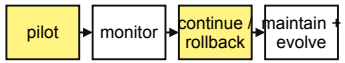
```
Normal + boundary + invalid + unauthorised + replay/abuse + failure/recovery.
```

TRAP *Test count and code coverage do not prove important behaviour or threat resistance.*

12 · Release + future change

C11

Deployment is controlled release into an operating environment. ***Maintenance*** includes corrective, adaptive, perfective and preventive change. ***Technical debt*** is future cost or constraint created by a current shortcut or deferred quality work.



Operational evidence controls release.

PLAN BEFORE RELEASE	EVIDENCE
Dependencies + migration	Reconciliation and interface checks
Pilot + rollback criteria	Named thresholds, not date alone
Monitoring + support owner	Signals, response and escalation
Deferred work	Debt, consequence, trigger, owner

MAINTENANCE	QUESTION
Corrective	What fault is repaired?
Adaptive	What environment or rule changed?

FIT2001 Systems Development · Self-test drills

FIVE WORKED ANSWER PATHS · JUDGEMENT CARDS · CLASSIC TRAPS · EXAM-MORNING RETRIEVAL

EXAM REVISION SHEET
Side 2 · drills + traps
Answer before revealing

Full guide + ask Sia about FIT2001

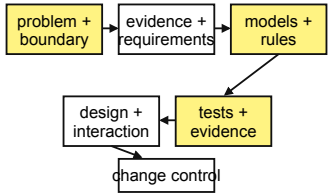
SIDE 2/2 COVER THE ANSWER · WRITE THE PATH · CHECK THE TRAP · RETRY WITH A CHANGED CASE

SELF-TEST, THEN CORRECT

Five worked mini-examples · source values preserved

13 · Method chooser C1–C11

Do not begin with a memorised diagram. Classify the task, name the artefact, state the decision it supports, then make the conclusion change when evidence changes.



Keep each artefact traceable to the problem.

IF PROMPT ASKS...	USE / SHOW
Frame scope/value	Boundary, actors, evidence, measures
Choose approach	Method fit by risk/uncertainty
Discover needs	Triangulated elicitation
Specify behaviour	Story + criteria + activity; use case
Represent concepts	Classes, associations, multiplicities, rules
Evaluate interface	User/task/context + observed evidence
Design collaboration	Responsibilities + ordered messages
Prove quality/security	Threat, oracle, trace row
Release/evolve	Migration, pilot, monitor, rollback, debt

PROMPT VERBS → ANSWER SHAPE

VERB	MINIMUM COMPLETE RESPONSE
Define	Distinct meaning + boundary from neighbouring term
Identify	Name case evidence and classify it
Compare	Same criteria, consequences and fit
Model	Elements + relations + constraints + meaning
Recommend	Decision + conditions + trade-off + evidence
Evaluate	Criterion + observed evidence + judgement + revision
Test	Input/state + action + oracle + trace + residual risk

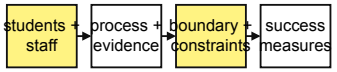
CHANGED-CASE DRILL

Take any answer and change one fact: actor, rule, multiplicity, exception, accessibility need, external-service failure or threat capability. Identify the first artefact that must change, then trace the change forward. If only the conclusion changes and no intermediate artefact changes, the original trace was probably asserted rather than reasoned.

CHECK EVERY ARROW	QUESTION
Evidence → requirement	What observation or rule supports it?

14 · Worked 1 · Library queue C1 · 4 MARKS

Question. A team proposes an app because students dislike queues. What must be defined before features?



Four marks before any feature choice.

1. Name affected students and staff.
2. Describe the current process and evidence.
3. Set the system boundary and constraints.
4. Choose success measures independent of the app idea.

STEP	WHY IT EARNS THE MARK
Actors	Replaces generic "users" with affected goals and work.
Current process	Separates observed causes from the proposed solution.
Boundary	Exposes responsibilities, constraints and dependencies.
Measures	Allows alternatives to be judged against the same value.

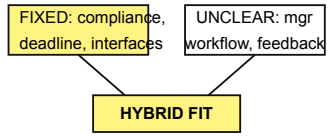
ANSWER

```
Define queue causes, actors, process
and measurable outcomes first; the app
is one possible intervention, not the
problem statement.
```

TRAP *Do not let the proposed interface become the system boundary.*

15 · Worked 2 · Payroll C2 · 4 MARKS

Question. A payroll replacement has fixed statutory deadlines but unclear manager workflows. Recommend an approach.



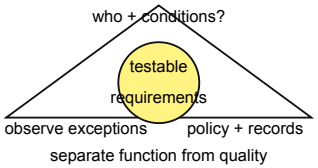
Control fixed risk; iterate uncertainty.

1. Separate fixed compliance from uncertain workflow.
2. Identify integration and release risk.
3. Use staged controls for mandatory interfaces.
4. Iterate manager workflow with representative users.

FACT	METHOD CONSEQUENCE
Fixed statutory deadline	Plan compliance work and decision points.
Mandatory interfaces	Baseline integration constraints; test early.

16 · Worked 3 · Returns C3 · 4 MARKS

Question. Staff say returns are "usually approved quickly". Turn that statement into a requirement investigation.



separate function from quality

Vague speed claims are starting evidence.

1. Ask who can approve and under which conditions.
2. Observe exceptions and actual system steps.
3. Inspect policy and transaction records.
4. Separate functional approval from time, security and audit qualities.

ANSWER

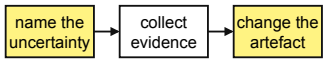
```
Triangulate staff accounts, observed
exceptions and policy; then write
testable approval, timing, access and
audit requirements.
```

TRAP *Elicitation should expose disagreement before the system encodes it.*

17 · Judgement cards C4 · C5 · C7 · C8

IF YOU SEE X...	DO Y
"As a user… know things"	Specify decision-maker, decision, data, timing, access and exceptions.
Use case = clicks only	Recover actor goal, system responses, pre/postconditions and attached alternatives.
Prototype shows success only	Target disputed/risky requirement; model rejection, missing evidence, escalation.
Form clears valid fields	Preserve input, explain specific error, focus accessibly, enable immediate recovery.
Diagram "looks complete"	Ask which requirement, decision, exception and test each element supports.

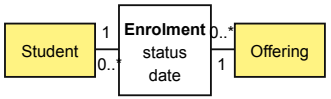
CHANGED FACT	FIRST ARTEFACT TO REVISIT
New user role/goal	Stakeholder evidence, story and use case
New denial/exception	Criteria, activity and alternate flow
New regulatory constraint	Non-functional requirement and method fit



Observation must change an artefact.

18 · Worked 4 · Enrolment C6 · 4 MARKS

Question. A student can enrol in many offerings and each offering has many students, with status and date per enrolment. Build the core model.



unique active pair: Student + Offering

Relationship data sits on Enrolment.

1. Identify Student and Offering as domain classes.
2. Create Enrolment as an association class.
3. Place status and date on Enrolment.
4. State multiplicities and uniqueness rules.

VALIDATION QUESTION	EXPECTED RESULT
Whose status/date?	One value per Student–Offering relationship
Can Student exist unenrolled?	Test the stated minimum at that end
Can Offering have many students?	Maximum permits many Enrolments
Duplicate active pair?	Prevent through an explicit uniqueness rule

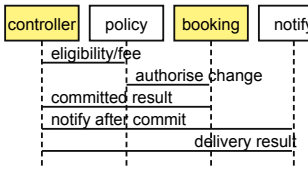
ANSWER

```
Student and Offering connect through
Enrolment, which carries status and
date; multiplicities and a uniqueness
rule prevent duplicate active
enrolments.
```

TRAP *Attributes belong on the concept whose one instance owns one value.*

19 · Realisation judgement C9

Cancellation case. Fee rule + notification. Start from preconditions/alternatives; assign booking, policy and notification; order validation, state change and message.



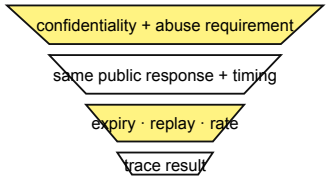
State owner changes; notify after commit.

MODEL ANSWER

```
Controller coordinates; policy
calculates eligibility/fee; booking
owns transition; notification
communicates only after committed
result.
```

20 · Worked 5 · Account reset C10 · 4 MARKS

Question. A reset flow passes normal tests but leaks whether an email is registered. Build a better test set.



Adversarial oracles test non-disclosure.

1. State confidentiality and abuse requirements.
2. Create registered and unregistered cases with the same public response.
3. Test rate, token expiry and replay.
4. Trace every result to evidence and residual risk.

CASE	ORACLE
Registered email	Public response does not reveal existence
Unregistered email	Same public response and timing expectation
Expired token	Rejected; no account change
Replayed token	Single-use control rejects reuse
Repeated requests	Rate control activates without disclosure

ANSWER

```
Public response and timing must not
disclose account existence; valid
tokens are single-use, expiring and
rate-controlled; tests cover misuse as
well as success.
```

TRAP *A security requirement needs an explicit adversarial oracle.*

21 · Classic traps C1–C11

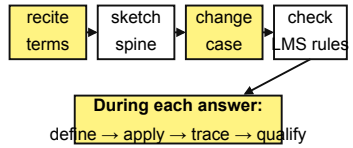
WEAK MOVE	REPAIR
Choose technology before problem	Boundary + evidence + outcome first
Call method "agile" as justification	Name uncertainty, risk, controls
Copy one stakeholder statement	Triangulate and surface conflict
Story without conditions	Add criteria + alternate paths
Use case without exceptions	Attach alternate to a main step
Domain model copies screens/data	Recover concepts + business rules
Polished prototype = validation	State question and changed decision
Familiar UI = usable	Observe specified users in context
One giant controller	Assign coherent domain responsibilities

TRACEABILITY SPOT-CHECK

22 · Fast recall grid GLOSSARY

TERM	RECITE IN ONE BREATH
Boundary	Controlled system vs environment/external actor
Stakeholder	Affects or is affected by system/outcomes
Method fit	Approach aligned to conditions
Functional req.	Capability/behaviour under conditions
Non-functional req.	Quality, constraint, operating condition
Acceptance criterion	Observable satisfactory condition
Alternate flow	Deviation/exception tied to main step
Multiplicity	Instance-count constraint at association end
Prototype	Incomplete representation for learning
Usability	Effectiveness, efficiency, satisfaction in context
Responsibility	Coherent class/component obligation
Test oracle	Expected outcome judging observed behaviour
Technical debt	Future cost from shortcut/deferred quality

23 · Exam-morning plan COURSE · BRIEF



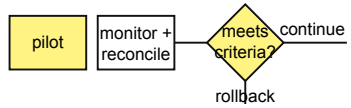
Retrieve, sketch, change and correct.

1. **Recite** definitions without notes; distinguish neighbouring terms.
2. **Sketch** the universal answer spine and the nine artefact choices.
3. **Redo** one worked case after changing an actor, rule, exception or risk.
4. **Check** the LMS exam instructions and personal timetable; the materials status is not confirmed here.

ANSWER DISCIPLINE

```
1. Underline task verb + facts. 2. Define
chosen concept. 3. Apply every fact. 4.
Trace to next artefact. 5. Test
exception/limit. 6. Conclude in the case
language.
```

24 · Release judgement C11



Evidence decides continue or rollback.