Domanda (punti 8/30)

Consolidate pratiche di gestione di progetto (*project management*) consigliano l'uso del *consuntivo di periodo*, anche detto "retrospettiva", come strumento essenziale per misurare il grado di avanzamento e valutare le corrispondenti prospettive di completamento. Precondizione essenziale all'efficace attuazione di tale pratica è il disporre di una interpretazione "intelligente" (non superficiale) di cosa essa sia. Questa prova d'esame vi chiede di:

1. fornire una definizione di "consuntivo di periodo", sia come nozione in sé che in termini dei dati che esso racchiude e della specifica funzione informativa di ciascuno di essi, indicando da quali fonti essa sia tratta o derivata;
2. spiegare quali indicazioni operative, utili alla gestione di progetto, possano essere tratte da un consuntivo siffatto;
3. suggerire una metodica utile ad alimentare una raccolta dati sostenibile che alimenti il consuntivo di periodo.

Procedo con ricerca nel materiale di progetto per validare e raffinare la risposta tecnica.## Risposta strutturata alla domanda d'esame sul consuntivo di periodo

Premessa metodologica: La risposta integra ancoraggio normativo (ISO/IEC 12207, CMMI, Scrum), definizioni operative, contrasti differenziali e esempi concreti dal progetto didattico per costruire una comprensione "intelligente" del consuntivo di periodo.

---

1. Definizione di "consuntivo di periodo"

Ancoraggio normativo e definizione

Il **consuntivo di periodo** (o **retrospettiva**) è il rendiconto quantitativo e qualitativo dei risultati ottenuti in un intervallo temporale predefinito di un'attività progettuale.

Fonti normative:

- **ISO/IEC 12207** (Software Life Cycle Processes): definisce i processi di Project Monitoring and Control che richiedono la misurazione sistematica dell'avanzamento contro il piano
- **CMMI** (Capability Maturity Model Integration): area di processo "Measurement and Analysis" specifica che "information needs are based on goals, constraints, risks, and problems which originate from the technical and management processes"
- **Scrum Framework**: la "Sprint Retrospective" struttura il consuntivo attorno a: (1) "Cosa abbiamo imparato nel periodo precedente?" (2) "Cosa significa questo per il futuro?"
- **ISO/IEC 15939** (Software Measurement Process): fornisce il modello dei processi di misurazione per supportare le decisioni di progetto

Nozione in sé

Il consuntivo di periodo opera in **modalità push**: il completamento delle attività causa notifica senza necessità di interrogazione esplicita. **Diversamente da** un semplice report contabile, il consuntivo è strumento di *apprendimento organizzativo* che trasforma dati grezzi in conoscenza operativa.

La definizione si estende dalla pratica agile della retrospettiva, che richiede analisi non solo *quantitativa* (costi, tempi) ma *qualitativa* (problemi emersi, apprendimenti, azioni correttive). Come indicato nei materiali

didattici (T04 Gestione di progetto), il consuntivo "deve catturare il rapporto tra l'avanzamento conseguito e il consumo riscontrato".

Dati racchiusi e funzione informativa

Il consuntivo aggrega dati eterogenei con specifiche funzioni informative:

Dato	Fonte	Funzione Informativa	Misurabilità
Baseline raggiunte	Repository di configurazione (CI versionati)	Quantificazione oggettiva dell'avanzamento tecnico. Le baseline sono "formally approved version of a configuration item, fixed at a specific time" (ISO 12207), rappresentano "punti di arrivo tecnico dai quali non si retrocede".	Tempestiva, accurata, non-intrusiva tramite controllo versione
Milestone completate	Diagrammi PERT/Gantt, calendario progetto	Valutazione temporale: verifica corrispondenza tra tempistiche pianificate e realizzate. Ogni milestone "denota un punto di avanzamento atteso, sostanziato da una baseline corrispondente". Nel progetto didattico: RTB (Requirements and Technology Baseline), PB (Product Baseline), CA (Customer Acceptance).	Misurabile per data e criteri di completamento (Definition of Done)
Risorse consumate	Timesheet, strumenti project tracking	Misurazione efficienza produttiva tramite metriche quali produttività (rapporto output/risorse) e tempo-persona effettivamente allocato. **Diversamente da** ore di orologio, le ore produttive misurano il tasso di raggiungimento obiettivi.	Accurata se integrata nel workflow (no overhead manuale)
Costi effettivi	Rendicontazione finanziaria, sistemi contabili	Confronto con preventivo iniziale per calcolare varianze (scostamenti) e identificare aree di inefficienza economica. Alimenta il **preventivo a finire (PAF)**.	Tempestiva per supportare decisioni di riallocazione
Indicatori di qualità	Piano di Qualifica, attività V&V	Verifica che il lavoro prodotto soddisfi criteri di **efficacia** (raggiungimento obiettivi) ed **efficienza** (ottimizzazione risorse). La combinazione determina l'**economicità** del processo.	Proattiva (non reattiva) tramite metriche automatizzate (es. coverage test, complexity metrics)
Problemi emersi e risoluzioni	Issue tracking, Problem Resolution (ISO 12207)	Analisi retrospettiva degli impedimenti e azioni correttive implementate, alimentando apprendimento organizzativo. Ogni problema "va studiato individualmente, decidendo soluzioni accettabili, realizzandole e verificando l'esito".	Non-intrusiva se integrata in workflow Agile (daily standup, sprint review)
Modifiche alla	Controllo versione	Tracking evoluzioni CI, permette di	Automatica tramite

Dato	Fonte	Funzione Informativa	Misurabilità
configurazione	e configurazione	quantificare **volatilità requisiti** (Requirement Stability Index) e impatto cambiamenti.	repository centralizzato

Nota critica: Una baseline deve essere *misurabile* secondo i criteri: **tempestivamente** (vicino al momento di creazione), **accuratamente** (senza ambiguità), **non-intrusivamente** (come sottoprodotto del lavoro normale, non attività aggiuntiva).

---

2. Indicazioni operative per la gestione di progetto

Il consuntivo genera indicazioni operative stratificate su livelli decisionali differenti:

Livello tattico (breve termine)

a) Ricalibrazione del Preventivo a Finire (PAF)

Il confronto consuntivo vs. preventivo rivela scostamenti che richiedono aggiornamento delle stime residue. Come specificato nei materiali didattici:

- Il PAF "deve diventare sempre più preciso col passare del tempo" incorporando apprendimenti reali
- **VINCOLO CRITICO**: "Il preventivo non si può mai aumentare, ma solo decrementare" (vincolo contrattuale con committente)
- **Diversamente da** un semplice aggiustamento contabile, il PAF richiede ripianificazione adattativa: "ogni azione in questo periodo chiede nuova pianificazione sul rimanente"

Esempio dal progetto didattico: Se il consuntivo del primo sprint rivela velocity effettiva di 15 story points vs. 20 pianificati, il PAF deve ricalcolare tutti gli sprint futuri assumendo velocity 15, eventualmente riducendo scope o estendendo durata (entro limiti contrattuali).

b) Gestione dei cammini critici

L'analisi PERT evidenzia se attività su percorsi critici (slack time = 0) hanno subito ritardi. Questo richiede:

- Immediate riallocazioni di risorse per accelerare attività critiche
- Revisione delle dipendenze temporali (possibile parallelizzazione?)
- Fast-tracking o crashing delle attività critiche se sostenibile economicamente

c) Ottimizzazione allocativa

Le metriche di produttività per risorsa/team identificano:

- **Colli di bottiglia**: risorse sovraccariche che rallentano intero progetto
- **Capacità residua**: risorse sottoutilizzate che possono assorbire task critici
- **Skill mismatch**: ruoli assegnati non allineati con competenze effettive

Esempio concreto: Se il consuntivo mostra che attività di verifica richiedono sistematicamente +30% tempo previsto, il PAF deve aumentare allocazione a ruolo Verificatore o rivedere Definition of Done.

Livello strategico (medio-lungo termine)

a) Ripianificazione adattativa

Come indicato nel modello agile: "La ripartizione del tempo di progetto in brevi periodi, con obiettivi e risorse contingentati, ognuno dei quali culminante in una retrospettiva, serve ad attenuare i rischi e favorire il miglioramento continuo".

Il consuntivo informa:

- **Backward planning** dalle milestone future: se RTB è raggiunta con 2 settimane ritardo, PB va ripianificata di conseguenza
- **Adattamento scope**: se velocity sistematicamente inferiore al previsto, negoziare con committente riduzione scope per rispettare deadline
- **Revisione incrementale**: decidere se procedere a incremento successivo o consolidare baseline corrente

b) Gestione proattiva dei rischi

Pattern ricorrenti nel consuntivo segnalano **rischi sistemici** vs. incidenti isolati:

- Se in 3 sprint consecutivi emerge problema "integration failure", è rischio architetturale che richiede refactoring
- Se velocity cala progressivamente, è segnale di **debito tecnico** accumulato o **burnout** del team

Azioni:

- Aggiornamento registro rischi con nuove categorie identificate
- Implementazione piani di contingenza (es. PoC per validare architettura alternativa)
- Allocazione buffer temporale/economico per rischi sistematici

c) Miglioramento processi (PDCA)

La retrospettiva identifica inefficienze strutturali nel *way of working*:

- **Diverse da** problemi puntuali (che richiedono azioni correttive immediate), le inefficienze di processo richiedono modifiche alle **Norme di Progetto**
- Ciclo Plan-Do-Check-Act: pianificare miglioramento → implementarlo in sprint successivo → verificare efficacia in retrospettiva → standardizzare se efficace

Esempio dal progetto didattico: Se retrospettiva rivela che code review richiede 3 giorni (troppo), la norma "max 24h per review" viene introdotta e monitorata nel consuntivo successivo.

Livello di comunicazione e governance

a) Trasparenza con stakeholder

Il consuntivo fornisce **evidenze oggettive** (baseline verificate, test coverage, burn-down chart) per dimostrazioni di progresso al committente. **Diversamente da** dichiarazioni soggettive ("siamo al 70%"), le baseline offrono:

- Artefatti eseguibili/testabili nel caso di RTB (PoC)
- Prodotti usabili nel caso di PB (MVP tra "usable" e "ready" secondo SEMAT)

Questo è essenziale in contesti contrattuali dove **pagamenti sono vincolati a milestone** (ad es. 30% alla RTB, 60% alla PB, 10% alla CA).

b) Decision making basato su dati

Metriche quantitative sostituiscono valutazioni soggettive:

- **Velocity** (contesti agili): story points completati per sprint
- **Earned Value Management** (approcci tradizionali): Budget Cost of Work Performed vs. Budget Cost of Work Scheduled
- **Lead Time**: tempo da apertura issue a chiusura con successo

Questo migliora qualità decisioni manageriali evitando bias cognitivi (sunk cost fallacy, optimism bias).

c) Controllo baseline scope

Tracking modifiche CI rivela se progetto subisce **scope creep** (aggiunta requisiti non concordati). Il consuntivo quantifica:

- **Requirement Stability Index**: quanti requisiti cambiano per periodo
- **Change Request Rate**: frequenza richieste modifica
- **Impact on schedule**: ritardo indotto da ogni change

Informa decisioni su **change control**: accettare/rifiutare modifiche, rinegoziare contratto, applicare Change Request formale.

---

3. Metodica per raccolta dati sostenibile

La raccolta dati deve essere **tempestiva, accurata e non-intrusiva** (principi di misurabilità delle baseline). La metodica si articola in tre livelli complementari:

A. Infrastruttura tecnologica

1. Repository centralizzato con controllo di versione

Componenti:

- Ogni Configuration Item (CI) ha identità unica: ID, nome, data, autore, registro modifiche
- Operazioni check-in/commit generano automaticamente metadati temporali e autoriali
- Sistema CI/CD (Continuous Integration/Continuous Deployment) produce metriche di build/test automaticamente

Sostenibilità: Dati generati come **sottoprodotto del workflow normale**, zero overhead aggiuntivo. Il developer non "compila il consuntivo", semplicemente lavora e il sistema cattura metriche.

Esempio concreto dal progetto didattico:

```
git commit -m "Fix: authentication timeout (#142)"
```

```
→ automaticamente traccia: autore, timestamp, issue collegata, LOC modificate
```

```
→ CI pipeline esegue: build, test unit, coverage analysis
```

```
→ Dashboard aggiorna: velocity, technical debt, defect density
```

2. Strumenti di project tracking integrati

Componenti:

- Issue tracking collegato al repository (es. Jira, Azure DevOps, GitHub Projects)
- Tracciamento automatico stato attività: Backlog → In Progress → Done
- Timesheet integrato per rilevazione tempo-persona con granularità task-level
- Link bidirezionale: issue ↔ commit ↔ pull request ↔ test results

Sostenibilità: **Single point of entry**, evita duplicazione informazioni. Il team aggiorna stato issue, non "compila report".

Esempio: Issue #142 passa a "Done" quando pull request è mergiata E tutti test passano E code review approvata → questo aggiorna automaticamente burn-down chart e velocity.

3. Dashboard di metriche in tempo reale

Componenti:

- Visualizzazioni automatiche: burn-down charts, velocity trend, test coverage, defect density

- Alert su deviazioni significative: "velocity -30% rispetto a media ultimi 3 sprint"
- Drill-down: da metrica aggregata (es. velocity) a dettaglio granulare (quali user story, quali blockers)

Sostenibilità: Informazione **pull sempre disponibile**, elimina reporting manuale. Il PM non "richiede status update", semplicemente apre dashboard.

Tecnologie: SonarQube (qualità codice), Grafana (metriche custom), GitHub Insights, Azure DevOps Analytics.

B. Processo organizzativo

1. Cicli di feedback regolari e brevi

Struttura:

- **Daily standup** (15 min, modello Scrum): aggiornamenti micro su avanzamento e impedimenti. Ogni membro risponde: "Cosa ho fatto ieri? Cosa farò oggi? Quali blockers?"
- **Sprint review** (2-4 settimane): dimostrazione funzionalità completate a stakeholder, raccolta feedback
- **Sprint retrospective**: consuntivo strutturato con template: "Start doing | Stop doing | Continue doing"

Sostenibilità: **Ritmo cadenzato** previene accumulo di "debito informativo". **Diversamente da** report mensili/trimestrali (troppo infrequenti), cicli brevi permettono correzioni tempestive.

Esempio dal progetto didattico: Sprint 2 settimane → retrospettiva ogni 2 settimane → se emerge problema "test environment instabile", può essere risolto PRIMA dello sprint successivo, non dopo 3 mesi.

2. Definition of Done (DoD) preventiva

Componenti:

- Ogni attività/user story ha criteri espliciti di completamento definiti A PRIORI
- DoD checklist può includere: codice scritto, unit test passed (coverage >80%), integration test passed, code review approvata, documentazione aggiornata, demo a stakeholder completata

Sostenibilità: Elimina **ambiguità su cosa costituisce "avanzamento"**. Il raggiungimento DoD genera automaticamente evidenze per consuntivo (test report, review approval).

Diversamente da stime soggettive ("task completato al 90%"), DoD è binaria: soddisfatta o non soddisfatta.

Esempio: User story "Login utente" ha DoD:

```
[✓] Codice implementato
[✓] Unit test coverage >85%
[✓] Integration test passed
[✓] Security review approvata
[✓] UI/UX review approvata
[✓] Documentazione API aggiornata
→ Solo quando TUTTI criteri soddisfatti, story è "Done" e contribuisce a velocity
```

3. Responsabilizzazione distribuita

Principi:

- Ogni membro team aggiorna proprio stato senza intermediazione PM
- Peer review e quality gates richiedono documentazione contestuale
- **Ownership individuale** su metriche personali (es. ogni developer vede proprio code quality trend)

Sostenibilità: **Carico distribuito**, evita colli di bottiglia su singoli ruoli (PM non diventa "compilatore di report"). Crea accountability personale.

Esempio: Developer X vede che suo cyclomatic complexity medio è 15 (soglia 10) → self-organizza refactoring senza attendere direttiva PM.

C. Disciplina dei dati

1. Granularità appropriata

Principi:

- Raccolta a livello **task** (non troppo fine: no singole linee codice) e **milestone** (non troppo grossolano: no solo "progetto completato")
- Metriche allineate ai **bisogni informativi** (ISO/IEC 15939): "information needs are based on goals, constraints, risks, and problems"
- Evitare "data swamps": raccogliere solo metriche che informano decisioni

Esempio di granularità corretta:

- ✓ MISURARE: velocity per sprint, defect per modulo, coverage per componente
- ✗ NON MISURARE: LOC per singola funzione, commit count per developer (vanity metrics)

2. Automazione della derivazione

Principi:

- Metriche **composte** calcolate automaticamente da dati primari
- Esempio: `Produttività = LOC_effettive / Effort_ore` derivata da repository (LOC) e timesheet (effort)
- Esempio: `Technical Debt Ratio = Remediation_Cost / Development_Cost` derivata da SonarQube

Sostenibilità: Riduce **errori umani** e effort di consolidamento. Le metriche sono sempre aggiornate, non richiedono ricalcolo manuale.

3. Vincoli di integrità

Componenti:

- Schema dati standardizzato per CI (ogni CI ha stesso set di metadati)
- Validazione automatica: timesheet non può eccedere 24h/giorno, coverage non può essere >100%, effort non può essere negativo
- Tracciabilità: ogni dato ha timestamp, autore, audit trail

Sostenibilità: **Qualità dati by design**, prevenzione vs. correzione. I dati "cattivi" sono bloccati all'ingresso, non scoperti mesi dopo.

Esempio: Timesheet entry:

```
Date: 2025-01-15

Developer: Alice

Task: #142

Effort: 4h
```

```
Validation:
  ✓ Date <= today
  ✓ Total effort for Alice on 2025-01-15 <= 8h
  ✓ Task #142 exists and is "In Progress"
→ Entry accettata
```

Esempio di implementazione concreta (progetto Scrum 2 settimane)

Giornaliero:

- Automated build metrics da CI/CD pipeline: compile time, test pass rate, coverage
- Daily standup registrato in issue tracker (blockers → issue automaticamente creata)

Settimanale:

- Team lead consolida issue tracking status
- Identifica blockers sistemici vs. incidentali
- Aggiorna burn-down chart, verifica allineamento con sprint goal

Bi-settimanale (Sprint Retrospective):

- Consuntivo formale generato con:
  - **Velocity effettiva vs. pianificata** (story points completed)
  - **Burn-down chart** vs. baseline ideale
  - **Defect escape rate** (bug trovati in produzione vs. in sviluppo)
  - **Lista problemi emersi** + azioni correttive assegnate
  - **Aggiornamento PAF** basato su nuova velocity e rischi emersi
- Output: documento consuntivo + retrospettiva video registrata

Mensile:

- Consolidamento multi-sprint per reporting a committente
- Aggiornamento Piano di Progetto con calendario rivisto
- Dimostrazione baseline a stakeholder (RTB → PoC funzionante, PB → MVP usabile)
- Ricalcolo ROI basato su costi effettivi vs. valore consegnato

---

Conclusione metodologica

La combinazione di **infrastruttura adeguata** (repository, CI/CD, dashboard), **processo disciplinato** (DoD, cicli feedback brevi, responsabilizzazione distribuita) e **cultura data-driven** (metriche mirate, automazione, integrità dati) rende il consuntivo di periodo uno **strumento di governance leggero ma potente**.

Principio chiave: "Misurare tempestivamente, accuratamente e non-intrusivamente". I dati devono emergere come **sottoprodotto del lavoro quotidiano**, non come attività aggiuntiva. Questo allineamento con il *way of working* naturale del team è l'unica garanzia di sostenibilità nel lungo periodo.

Diversamente da approcci tradizionali dove il consuntivo è evento episodico e oneroso (compilazione manuale report mensili), l'approccio proposto rende il consuntivo **emanazione naturale e continua** del processo di sviluppo.

Implicazione in contesti safety-critical: In domini dove errori software hanno conseguenze life-threatening (es. sistemi medicali, aerospaziale, automotive), il consuntivo non è solo strumento di gestione ma **artefatto**

di compliance verso normative (IEC 62304 per medical devices, DO-178C per avionica). La tracciabilità end-to-end (requisiti → design → codice → test → deployment) e l'evidenza di rigore processuale (Problem Resolution, Configuration Management) sono prerequisiti certificativi. In tali contesti, la metodica proposta deve essere integrata con audit trail immutabili e revisioni indipendenti.

Fonti bibliografiche consolidate:

- ISO/IEC 12207:2017 - Systems and software engineering — Software life cycle processes
- ISO/IEC 15939:2017 - Systems and software engineering — Measurement process
- CMMI for Development, Version 1.3 - SEI Carnegie Mellon
- Scrum Guide 2020 - Ken Schwaber & Jeff Sutherland
- Materiali didattici T04 "Gestione di progetto" - Prof. Tullio Vardanega, UniPD
- Glossari corso Ingegneria del Software 2018-2024 - UniPD

BREVE

---

1. Definizione di "consuntivo di periodo"

Ancoraggio normativo

Il **consuntivo di periodo** (o **retrospettiva**) è il rendiconto quantitativo e qualitativo dei risultati ottenuti in un intervallo temporale predefinito di un'attività progettuale.

Fonti:

- **ISO/IEC 12207**: processi di Project Monitoring and Control richiedono misurazione sistematica dell'avanzamento
- **CMMI**: area "Measurement and Analysis" stabilisce che "information needs are based on goals, constraints, risks, and problems"
- **Scrum Framework**: Sprint Retrospective strutturata su: (1) "Cosa abbiamo imparato?" (2) "Cosa significa per il futuro?"
- **ISO/IEC 15939**: modello dei processi di misurazione

Nozione in sé

Opera in **modalità push**: il completamento attività genera notifica automatica. **Diversamente da** un semplice report contabile, il consuntivo è strumento di *apprendimento organizzativo* che trasforma dati in conoscenza operativa, catturando "il rapporto tra avanzamento conseguito e consumo riscontrato" (T04 Gestione di progetto).

Dati racchiusi e funzione informativa

Dato	Fonte	Funzione Informativa
Baseline raggiunte	Repository CI versionati	Quantificazione oggettiva avanzamento tecnico. "Formally approved version of configuration item, fixed at specific time" (ISO 12207). Punti di arrivo tecnico non retrocedibili.
Milestone completate	PERT/Gantt	Verifica corrispondenza tempistiche pianificate vs. realizzate. Ogni milestone sostanziata da baseline (RTB, PB, CA nel progetto didattico).
Risorse consumate	Timesheet, tracking tools	Efficienza produttiva: ore-persona effettive, produttività (output/risorse). **Diversamente da** ore orologio, le ore produttive misurano tasso raggiungimento obiettivi.
Costi effettivi	Rendicontazione finanziaria	Varianze vs. preventivo, alimenta PAF (Preventivo a Finire).
Indicatori qualità	Piano Qualifica, V&V	Efficacia (obiettivi raggiunti) ed efficienza (risorse ottimizzate) → economicità processo.
Problemi e risoluzioni	Issue tracking, Problem Resolution (ISO 12207)	Retrospettiva impedimenti e azioni correttive, apprendimento organizzativo.
Modifiche configurazione	Controllo versione	Volatilità requisiti (Requirement Stability Index), impatto cambiamenti.

Requisito critico: baseline misurabile **tempestivamente, accuratamente, non-intrusivamente** (sottoprodotto lavoro normale, non attività aggiuntiva).

---

2. Indicazioni operative per la gestione di progetto

Livello tattico (breve termine)

a) Ricalibrazione PAF

- Confronto consuntivo/preventivo → aggiornamento stime residue
- **VINCOLO**: "preventivo non si può mai aumentare, solo decrementare" (vincolo contrattuale)
- Ripianificazione adattativa: "ogni azione chiede nuova pianificazione sul rimanente"

Esempio: velocity effettiva 15 story points vs. 20 pianificati → PAF ricalcola sprint futuri assumendo 15, riducendo scope entro limiti contrattuali.

b) Gestione cammini critici

- Analisi PERT: attività con slack time = 0 in ritardo → riallocazione risorse immediate, revisione dipendenze temporali

c) Ottimizzazione allocativa

- Identificazione colli di bottiglia (risorse sovraccariche) e capacità residua (sottoutilizzate)
- Skill mismatch: ruoli non allineati con competenze

Esempio concreto: attività verifica richiedono sistematicamente +30% tempo → aumentare allocazione Verificatore o rivedere DoD.

Livello strategico (medio-lungo termine)

a) Ripianificazione adattativa

- Backward planning da milestone future
- Adattamento scope se velocity sistematicamente inferiore
- Decisione incremento successivo vs. consolidamento baseline corrente

b) Gestione proattiva rischi

- Pattern ricorrenti identificano rischi sistemici vs. incidenti isolati
- Esempio: 3 sprint con "integration failure" → rischio architetturale, richiede refactoring
- Aggiornamento registro rischi, piani contingenza, buffer per rischi sistemici

c) Miglioramento processi (PDCA)

- **Diverse da** azioni correttive puntuali, inefficienze processo richiedono modifiche Norme di Progetto
- Ciclo Plan-Do-Check-Act: pianificare → implementare → verificare → standardizzare

Esempio: code review richiede 3 giorni → norma "max 24h review" introdotta e monitorata.

Livello comunicazione e governance

a) Trasparenza stakeholder

- Evidenze oggettive (baseline verificate, test coverage, burn-down) vs. dichiarazioni soggettive
- Essenziale per pagamenti vincolati a milestone (30% RTB, 60% PB, 10% CA)

b) Decision making data-driven

- Velocity (Agile), Earned Value (tradizionale), Lead Time sostituiscono valutazioni soggettive
- Evita bias cognitivi (sunk cost fallacy, optimism bias)

c) Controllo scope

- Tracking modifiche CI → quantifica scope creep

- Informa change control: accettare/rifiutare modifiche, rinegoziare contratto

---

3. Metodica per raccolta dati sostenibile

Principio: **tempestiva, accurata, non-intrusiva**.

A. Infrastruttura tecnologica

1. Repository centralizzato + versioning

- CI con identità unica (ID, nome, data, autore, registro)
- Check-in/commit generano metadati automaticamente
- CI/CD produce metriche build/test automaticamente
- *Sostenibilità*: dati come sottoprodotto workflow, zero overhead

Esempio:

```
git commit → traccia autore, timestamp, issue, LOC

CI pipeline → build, test, coverage

Dashboard → aggiorna velocity, technical debt
```

2. Project tracking integrato

- Issue tracking ↔ repository (Jira, Azure DevOps, GitHub)
- Stato automatico: Backlog → In Progress → Done
- Timesheet integrato granularità task
- *Sostenibilità*: single point of entry, no duplicazione

3. Dashboard real-time

- Burn-down, velocity, coverage, defect density auto-visualizzate
- Alert deviazioni significative
- *Sostenibilità*: informazione pull, elimina reporting manuale

B. Processo organizzativo

1. Cicli feedback brevi

- **Daily standup** (15min): avanzamento micro, impedimenti
- **Sprint review** (2-4 settimane): demo funzionalità, feedback stakeholder
- **Sprint retrospective**: consuntivo strutturato "Start/Stop/Continue"
- *Sostenibilità*: ritmo cadenzato previene debito informativo

2. Definition of Done preventiva

- Criteri espliciti completamento definiti a priori
- DoD checklist: codice, test passed (>80% coverage), review, documentazione, demo
- *Sostenibilità*: elimina ambiguità avanzamento, genera evidenze automatiche
- DoD binaria: soddisfatta o non soddisfatta (no "90% completato")

3. Responsabilizzazione distribuita

- Team aggiorna stato senza intermediazione PM
- Peer review richiede documentazione contestuale
- *Sostenibilità*: carico distribuito, evita colli bottiglia, crea accountability

C. Disciplina dei dati

1. Granularità appropriata

- Livello task e milestone (non troppo fine/grossolano)
- Metriche allineate bisogni informativi (ISO/IEC 15939)
- Evitare data swamps di metriche inutilizzate

2. Automazione derivazione

- Metriche composte calcolate da dati primari
- Esempio: `Produttività = LOC/Effort` da repository + timesheet
- *Sostenibilità*: riduce errori, effort consolidamento

3. Vincoli integrità

- Schema standardizzato CI
- Validazione automatica (timesheet ≤24h/giorno, coverage ≤100%)
- *Sostenibilità*: qualità by design, prevenzione vs. correzione

Implementazione concreta (Scrum 2 settimane)

- **Giornaliero**: metriche build automatiche da CI/CD
- **Settimanale**: consolidamento issue status, identificazione blockers
- **Bi-settimanale**: retrospettiva formale con velocity, burn-down, defect rate, problemi+azioni, aggiornamento PAF
- **Mensile**: consolidamento multi-sprint per committente, aggiornamento Piano Progetto, dimostrazione baseline

---

Conclusione

Infrastruttura + processo disciplinato + cultura data-driven rendono il consuntivo strumento di governance leggero ma potente. **Chiave**: dati emergono come sottoprodotto lavoro quotidiano, non attività aggiuntiva. Questo allineamento con *way of working* naturale garantisce sostenibilità.

Diversamente da approcci tradizionali (consuntivo episodico manuale), questa metodica rende il consuntivo emanazione naturale e continua del processo sviluppo.

Implicazione safety-critical: in domini medicali/aerospaziale, consuntivo è artefatto compliance normativa (IEC 62304, DO-178C). Tracciabilità end-to-end e rigore processuale sono prerequisiti certificativi.