2. Ciclo di Vita e Processi di Sviluppo

come procedere nello sviluppo?

Andrea Polini

Ingegneria del Software
Corso di Laurea in Informatica

Sommario

Ciclo di Vita

Con Ciclo di vita ci si riferisce agli stati che un certo ente può attraversare dal suo concepimento fino alla sua dismissione.

Un processo di sviluppo software si occupa delle strategie da adottare per la gestione del ciclo di vita di un prodotto software. In particolare identifica quando e chi deve fare cosa per raggiungere un determinato obiettivo della gestione del ciclo di vita.

Process Activities

Tipicamente i differenti processi software si distinguono in base alla organizzazione delle differenti attività dello sviluppo (workflow perspective) e da documenti/modelli che queste attività producono (data-flow perspective):

- Comunicazione (definire cosa deve fare il sistema software)
- Pianificazione (valutare i rischi ed i costi)
- Modellazione (modellare il sistema e progettarne le funzionalità)
- Costruzione (implementare le scelte e verificarle)
- Deployment (dispiegare nell'ambiente di produzione)

Comunicazione

- Capire cosa il sistema deve fare
- Quali sono i vincoli a cui il sistema deve sottostare

- Differenti tipi di specifica
  - Rivolta al customer
  - Rivolta allo sviluppatore

Comunicazione

...continua

Ingegneria dei requisiti prevede quattro task principali:

- Studio di fattibilità (allo scopo di comprendere le funzionalità fondamentali che il sistema deve implementare)
- Elicitazione dei requisiti ed analisi
- Specifica dei Requisiti
- Validazione dei Requisiti (realistici? consistenti? completi?)

Modellazione

Derivare una descrizione:

- del software che deve essere sviluppato e della sua architettura
- dei dati che devono essere scambiati
- componenti facenti parte del sistema
- interfacce tra i vari elementi del sistema
- algoritmi utilizzati

Modellazione

Approcci al design

- Metodologie agili tendono a ridurre i modelli sviluppati nelle fasi
- Metodi strutturati tendono a sviluppare molti modelli usando notazioni grafiche(i.e. UML)

Modellazione

...continua

Tipici modelli in approcci strutturati:

- Modello ad oggetti
- Modello di sequenze
- Modello a transizione degli stati
- Modello strutturale
- Modello del flusso dei dati

Costruzione

Implementazione

Obiettivo di questa attività è quello di mettere in atto azioni volte alla scrittura del codice del sistema, in “conformità” a quanto stabilito dai modelli definiti nelle attività di modellazione.

OO Programming
Design patterns

Costruzione

Verifica e Validazione

Obiettivo è verificare che il sistema soddisfa i requisiti e le esigenze del committente/utente.
Tecniche tipicamente utilizzabili:

- Statiche (i.e. Ispezione del codice)
- Dinamiche (i.e. Testing)

Tipicamente testing strutturato su più fasi:

- Testing di componente o di unità
- Testing di integrazione
- Testing di sistema
- Testing di Accettazione (alpha testing)
- Beta testing

Testing vs. Debugging?

Manutenzione

Riguarda le attività che sono messe in atto sul software già rilasciato.

Tipi di evoluzione:

- correttivo
- adattivo
- perfettivo

In definitiva

L'ingegneria del software intende fornire, tra le altre cose, molte "liste della spesa" che stabiliscono cosa debba essere fatto al fine di raggiungere determinati obiettivi. L'ingegnere del software sceglie la "spesa" più adatta alle sue esigenze!!

Manutenzione

Riguarda le attività che sono messe in atto sul software già rilasciato.

Tipi di evoluzione:

- correttivo
- adattivo
- perfettivo

In definitiva

L'ingegneria del software intende fornire, tra le altre cose, molte "liste della spesa" che stabiliscono cosa debba essere fatto al fine di raggiungere determinati obiettivi. L'ingegnere del software sceglie la "spesa" più adatta alle sue esigenze!!

Sommario

Modelli di Processo

Un processo software è costituito da un insieme di attività che conducono alla realizzazione di un prodotto software

- Nessuna soluzione generale! Infatti molte organizzazioni sviluppano proprio processo di sviluppo
- Forte dipendenza dalla persone e dall'organizzazione interna

Principali modelli di processo discussi (Attenzione! sono solo categorie):

- Modello a Cascata (waterfall)
- Modelli Evolutivi
- Modelli Iterativi
- Component-based Software Engineering (CBSE)
- Sviluppo tramite metodi formali (cleanroom development, B method, Model Driven Development)

Nella realtà spesso processi utilizzati sono una commistione dei diversi modelli

Processo a Cascata

Primo processo di sviluppo apparso in letteratura (Royce 1970). Fasi dello sviluppo strutturate in accordo alle attività fondamentali:

- Analisi e definizione dei requisiti
- Progettazione del sistema e del software
- Implementazione e test di unità
- Integrazione e test di sistema
- Istallazione e mantenimento

Processo a Cascata

- le varie attività in realtà vengono ripetute diverse volte ma comunque ad un certo punto “congelate” - ogni attività fornisce manufatti alle fasi successive attraverso appositi documenti
- terminata definitivamente un’attività questa NON viene ulteriormente riconsiderata nel seguito
- in generale auspicabile sovrapposizione dei vari team (se assegnati sulle fasi) per passaggio di informazioni

Conseguenze principali:

- I requisiti vengono fissati ad un certo punto e mai più modificati
- Prematura decisione sui requisiti rende il processo “sordo” alle successive richieste di adattamento dei requisiti da parte del cliente.
- Processo fortemente documentato e guidato dal rilascio di documenti (document driven).

Visibilità alta o bassa? Timeliness?

Quando applicarlo?

Processo Evolutivo

Il prodotto viene sviluppato cercando dapprima di derivare un prototipo e poi per incrementi successivi fino a raggiungere un sistema soddisfacente

- Il prototipo intende supportare le attività di comunicazione
- fasi di interazione con l'utente
- Prototipo usa e getta (Throaway Prototyping) ... ma spesso diventa prototipo da far evolvere

Conseguenze principali:

- L'organizzazione del sistema tende ad essere poco strutturata a causa dei continui cambiamenti
- Induce probabili difficoltà in fasi di mantenimento

Visibilità alta o bassa? Timeliness?

Quando applicarlo?

"Do it twice" - *Baker*

Processo Evolutivo

Il prodotto viene sviluppato cercando dapprima di derivare un prototipo e poi per incrementi successivi fino a raggiungere un sistema soddisfacente

- Il prototipo intende supportare le attività di comunicazione
- fasi di interazione con l'utente
- Prototipo usa e getta (Throaway Prototyping) ... ma spesso diventa prototipo da far evolvere

Conseguenze principali:

- L'organizzazione del sistema tende ad essere poco strutturata a causa dei continui cambiamenti
- Induce probabili difficoltà in fasi di mantenimento

Visibilità alta o bassa? Timeliness?

Quando applicarlo?

"Do it twice" - *Baker*

Processi Iterativi

Cercano di fondere le strutture dei processi evolutivi e a cascata cercando di mantenerne le qualità positive rimuovendo quelle negative.

Basati sul concetto di iterazione (la specifica si sviluppa con il sistema):

- Rilascio incrementale
- Sviluppo a spirale

Rilascio incrementale

- Processo procede con tanti mini “waterfall” in sequenza
- Ad ogni iterazione viene rilasciato un sistema funzionante che potrà essere utilizzato dal cliente.
- Si pianificano iterazioni successive introducendo via via nuove funzionalità
- Una volta attivata un’iterazione deve essere terminata senza interferenze
- Ogni iterazione dovrebbe tendere ad aumentare il sistema di una porzione gestibile (20.000 LOC?)

Rilascio incrementale

vantaggi

Principali vantaggi:

- Particolarmente efficace se team di sviluppo piccolo
- clienti non devono aspettare rilascio finale prima di poter utilizzare il sistema
- Uso sul prototipo comporta scoperta, definizione e modifica di requisiti
- si riduce rischio di fallimento
- funzionalità più importanti maggiormente testate

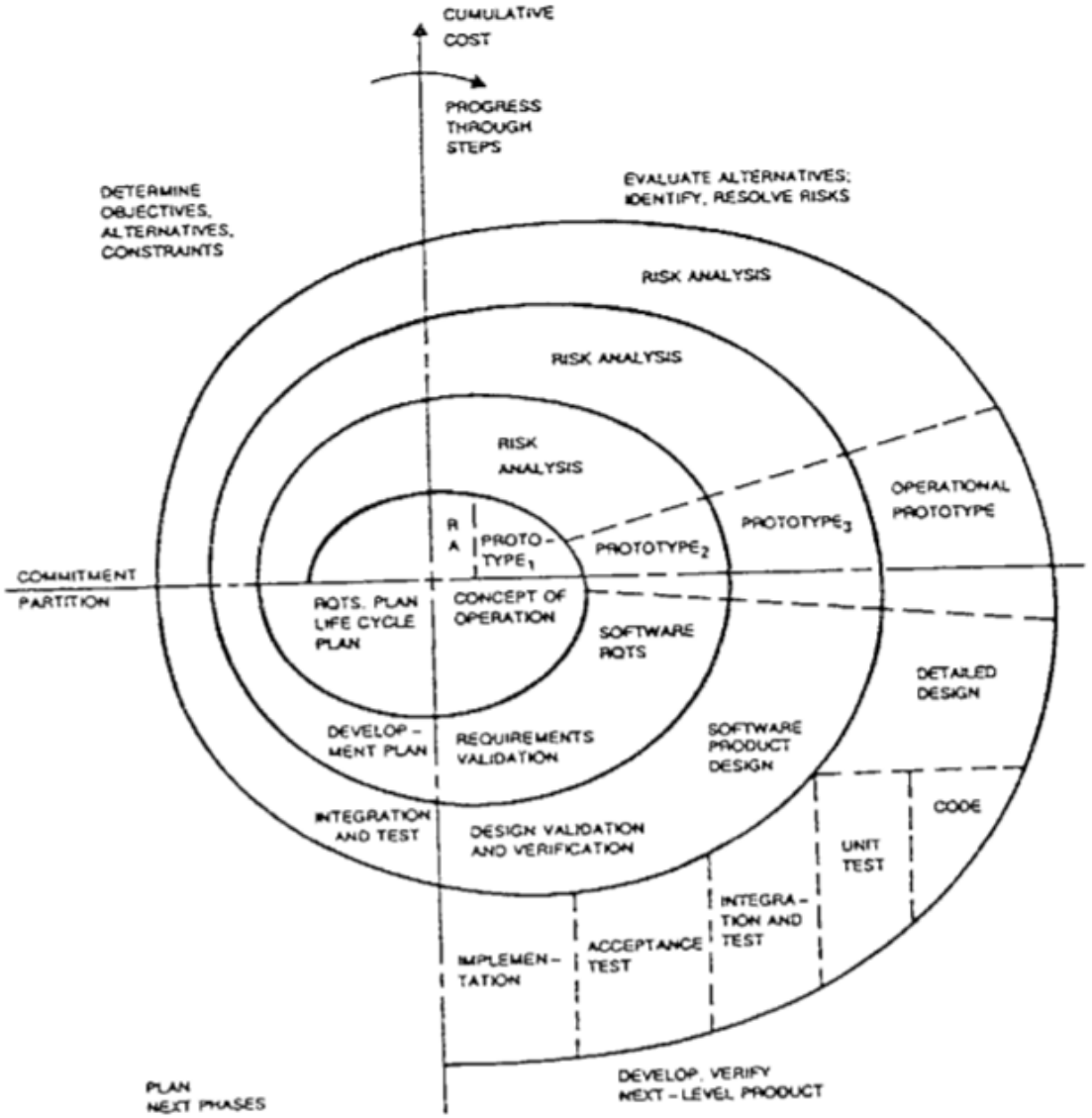
Sviluppo a Spirale

Proposto da Boehm nel 1988 introduce gestione del rischio nella pianificazione delle varie iterazioni.

Rappresentabile su un piano tramite una spirale dove ogni quadrante consiste di attività volte a:

- Specificare gli obiettivi dell'iterazione ed itentificazione dei rischi
- Valutare il rischio e definire techniche per la sua gestione
- Procedere allo sviluppo ed alla validazione
- Pianificazione, si procede a valutare la necessità di un'ulteriore iterazione.

Sviluppo a Spirale



Processi Iterativi

Visibilità alta o bassa? Timeliness?

Quando applicarlo?

Component based software engineering

generalità

Aumento nella complessità dei sistemi implica forte spinta al riuso (interno e non - COTS)

Obiettivo di CBSE: implementare un sistema come integrazione di componenti preconfezionati.

Un po' di storia . . .

- Conferenza di Garmisch 1968
- primi modelli a componenti OLE-X (anni ottanta)
- oggi . . .
  - Modelli di componenti Desktop (COM, Bonomo ...)
  - Modelli di componenenti distribuiti (CCM, EJB, ...)

Component based Software Engineering

Modelli di componenti

Definiscono una piattaforma di supporto al riuso ed una serie di regole per:

- interfacce da definire
- meccanismi di interazione tra componenti
- impacchettare il componente

Component based Software Engineering

Processo di sviluppo

Non si sono ancora evidenziati processi di sviluppo condivisi in questo ambito.

Principali differenze comunque si avranno con introduzione di nuove fasi nello sviluppo e nuove iterazioni.

- Component provisioning
- Requirements adaptation
- Design orientato al riuso
- Sviluppo orientato all'integrazione ed adattamento

Component based Software Engineering

Conseguenze

Visibilità alta o bassa? Timeliness? Produttività?

Quando applicarlo?

Approccio profondamente differente allo sviluppo. Introduzione all'interno di un'organizzazione richiede training e persone con forti doti di astrazione.

Capacità di sviluppare sistemi complessi in minor tempo. Riduce il rischio dello sviluppo. Tendenzialmente il prodotto finale dovrebbe essere "migliore".

Sviluppo di un semplice mailer di posta elettronica

Model Driven Development

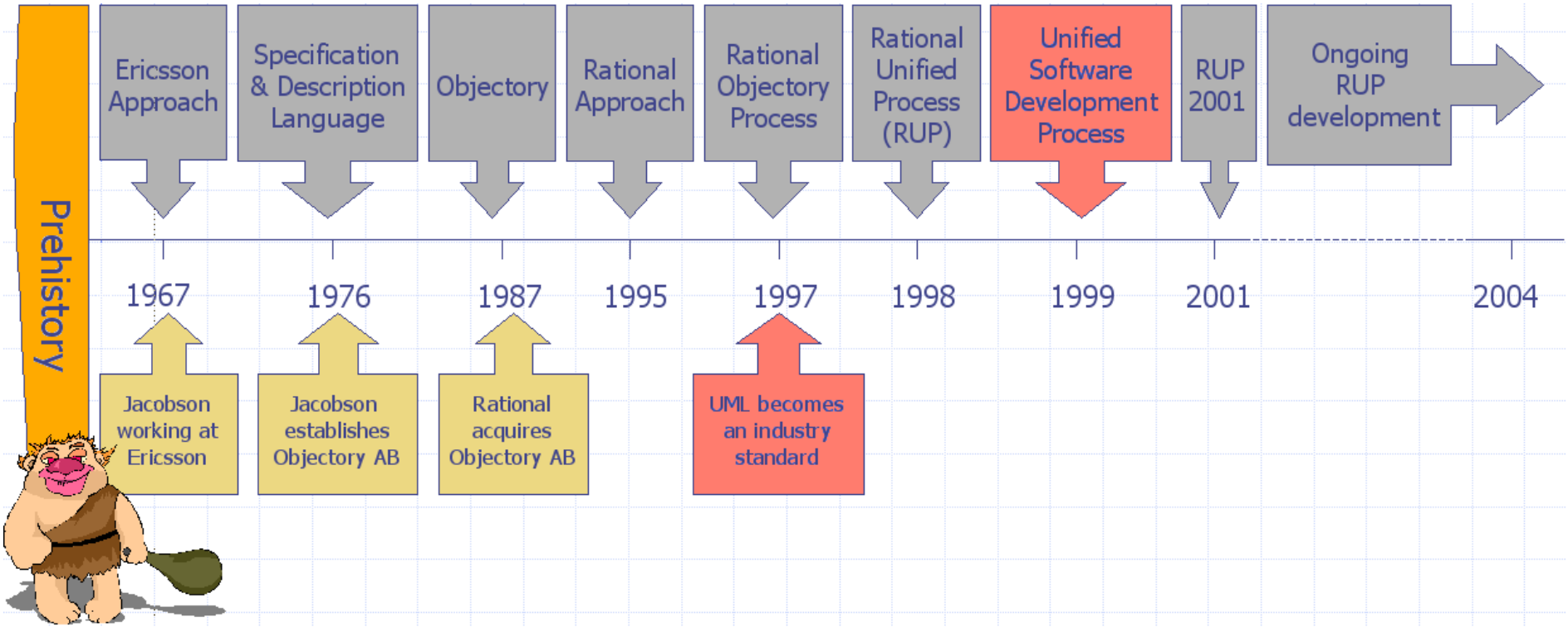
Modelli vengono definiti, fatti evolvere mantenendo consistenza ed aggiungendo informazioni via via di più basso livello fino ad arrivare al codice.

- CIM (modello dei requisiti)
- PIM (design di alto livello)
- PSM (design che tiene conto di dettagli della piattaforma di deployment)

Il Processo Unificato

- UP è un processo industriale “standardizzato” per lo sviluppo del software:
  - è il processo di riferimento che si associa all’uso di UML
  - è “free” – descritto nel libro “The Unified Software Development Process”
- Caratteristiche salienti:
  - Guidato dal rischio
  - Guidato dai Casi d’uso
  - È incentrato sull’architettura
  - Processo Iterativo incrementale
- UP è un processo generico di sviluppo del software. Deve essere adattato per ogni singolo progetto:
  - Standard interni, formati di documenti, tools, databases, modifiche al ciclo di vita . . .
- Rational Unified Process (RUP) è una particolare istanza di UP:
  - è un prodotto di Rational Corporation
  - Deve esssere adattato ma precisamente definito

Storia dello UP

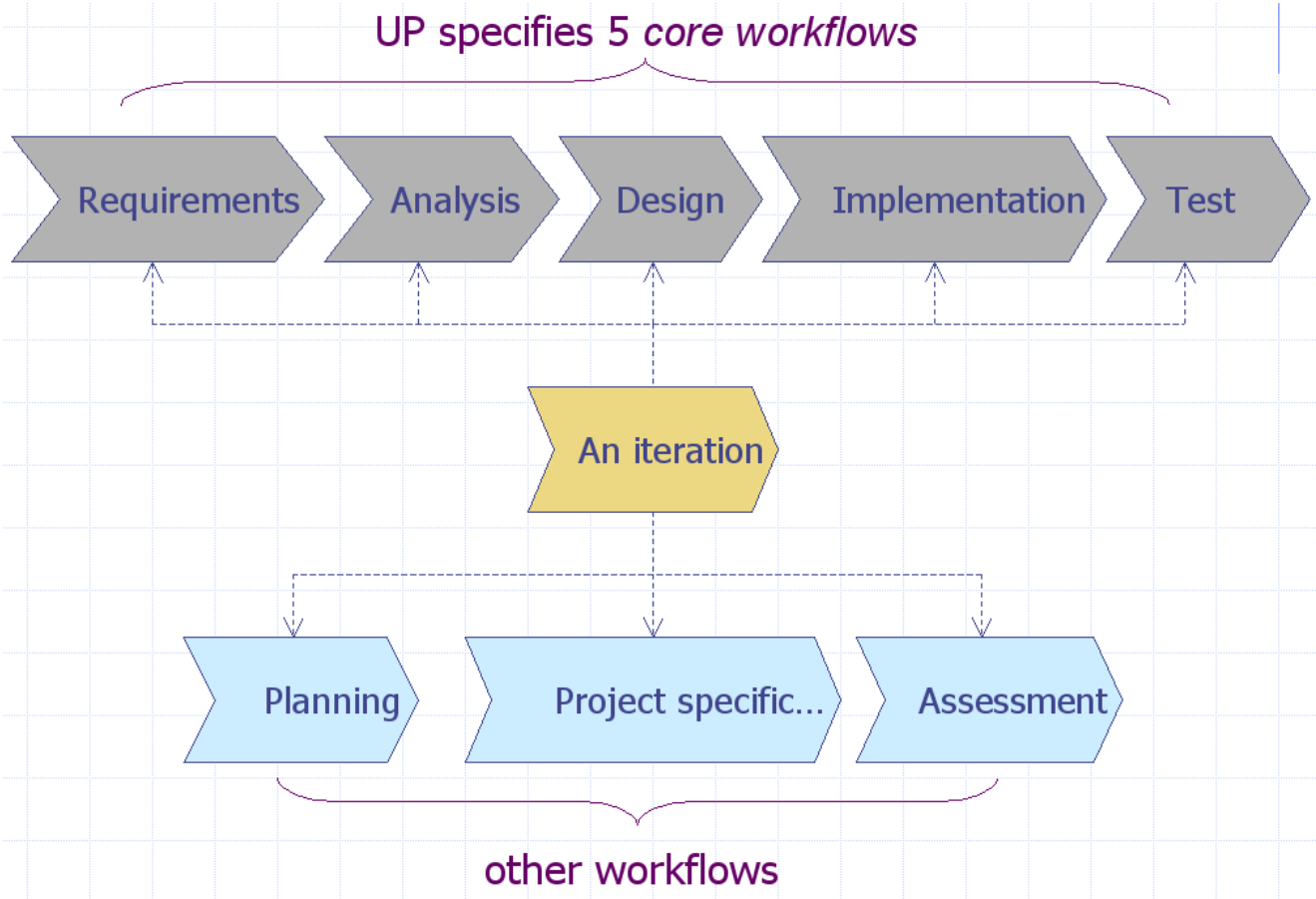


Iterazioni

- Un'iterazione può essere considerata come un mini progetto che include le seguenti attività:
  - Pianificazione
  - Requisiti
  - Analisi
  - Progettazione
  - Implementazione
  - Integrazione e Test
  - Validazione, Rilascio interno o esterno
- il prodotto viene derivato attraverso una sequenza di iterazioni
- le iterazioni si possono sovrapporre
- le iterazioni sono organizzate in fasi
- Favorire agilità: iterazioni "timeboxed" e durata ridotta (2-3 settimane)

Flusso di lavoro nelle iterazioni nello UP

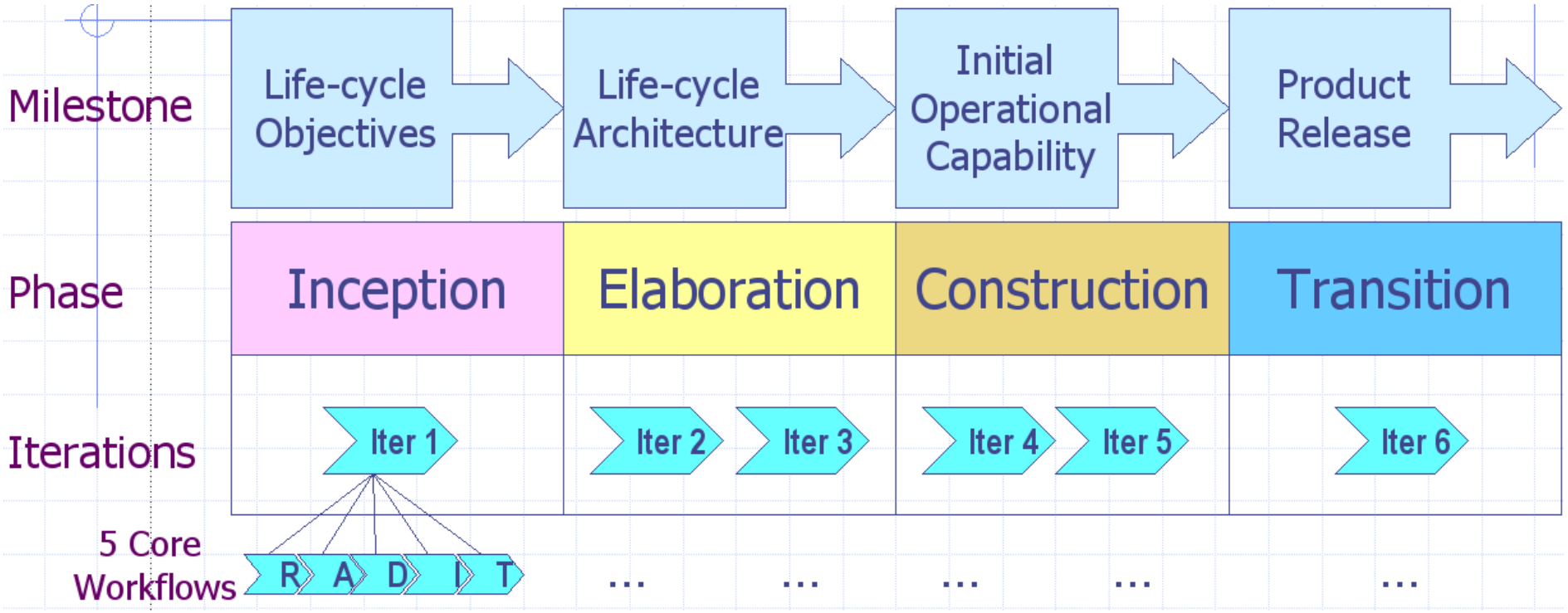
Ogni iterazione può contenere tutte le attività del flusso di lavoro con una differente enfasi su ognuna di esse in dipendenza dello stato di avanzamento del progetto



Baselines

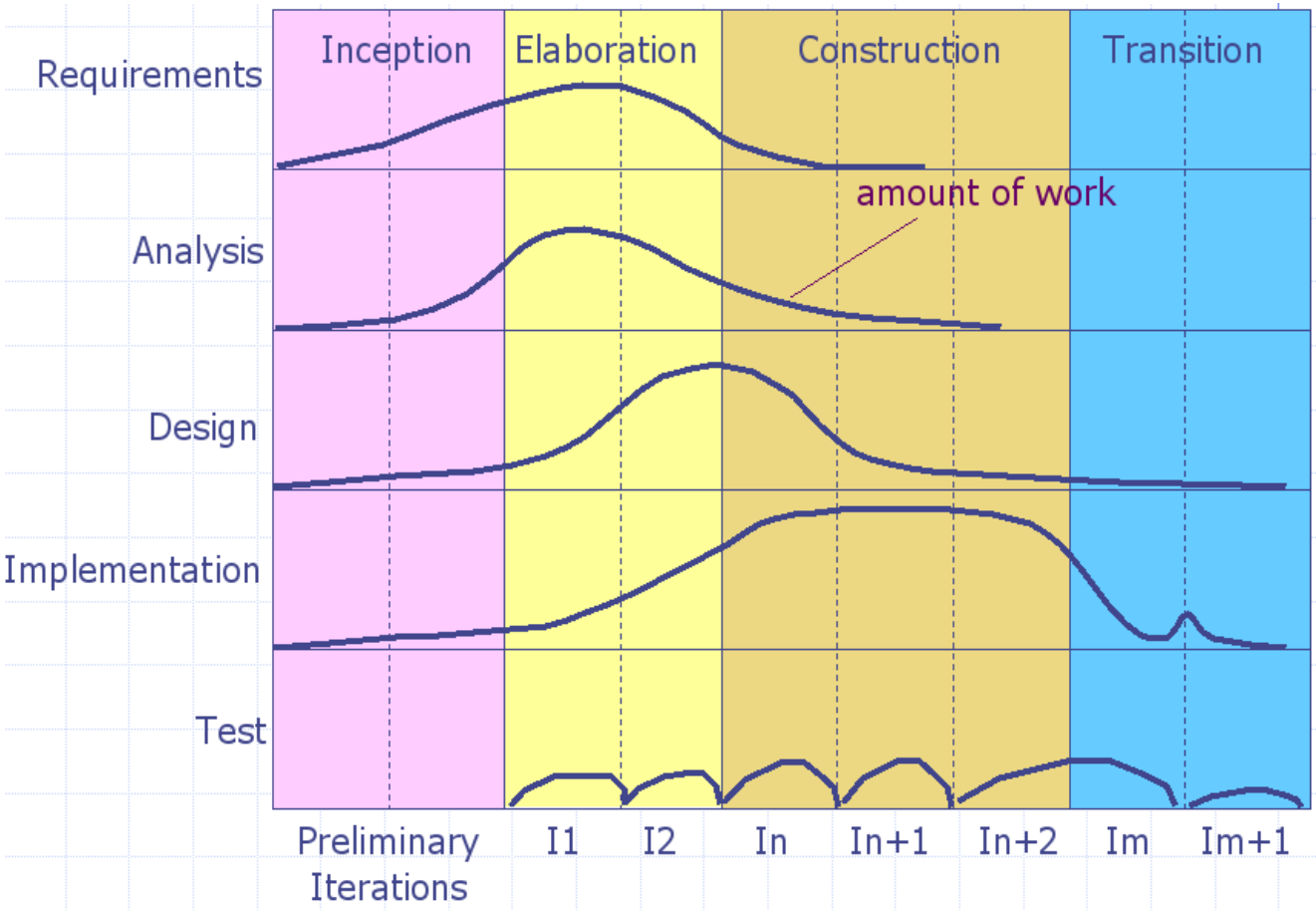
- Ogni iterazione genera una “baseline”
- una baseline è un insieme di artefatti rivisti ed approvati che:
  - Forniscono un base condivisa per ulteriore sviluppo
  - Possono essere modificati solo attraverso una procedura formale
- Un incremento è la differenza tra la baseline generata ad una iterazione e quella generata all’iterazione successiva

Struttura dello UP



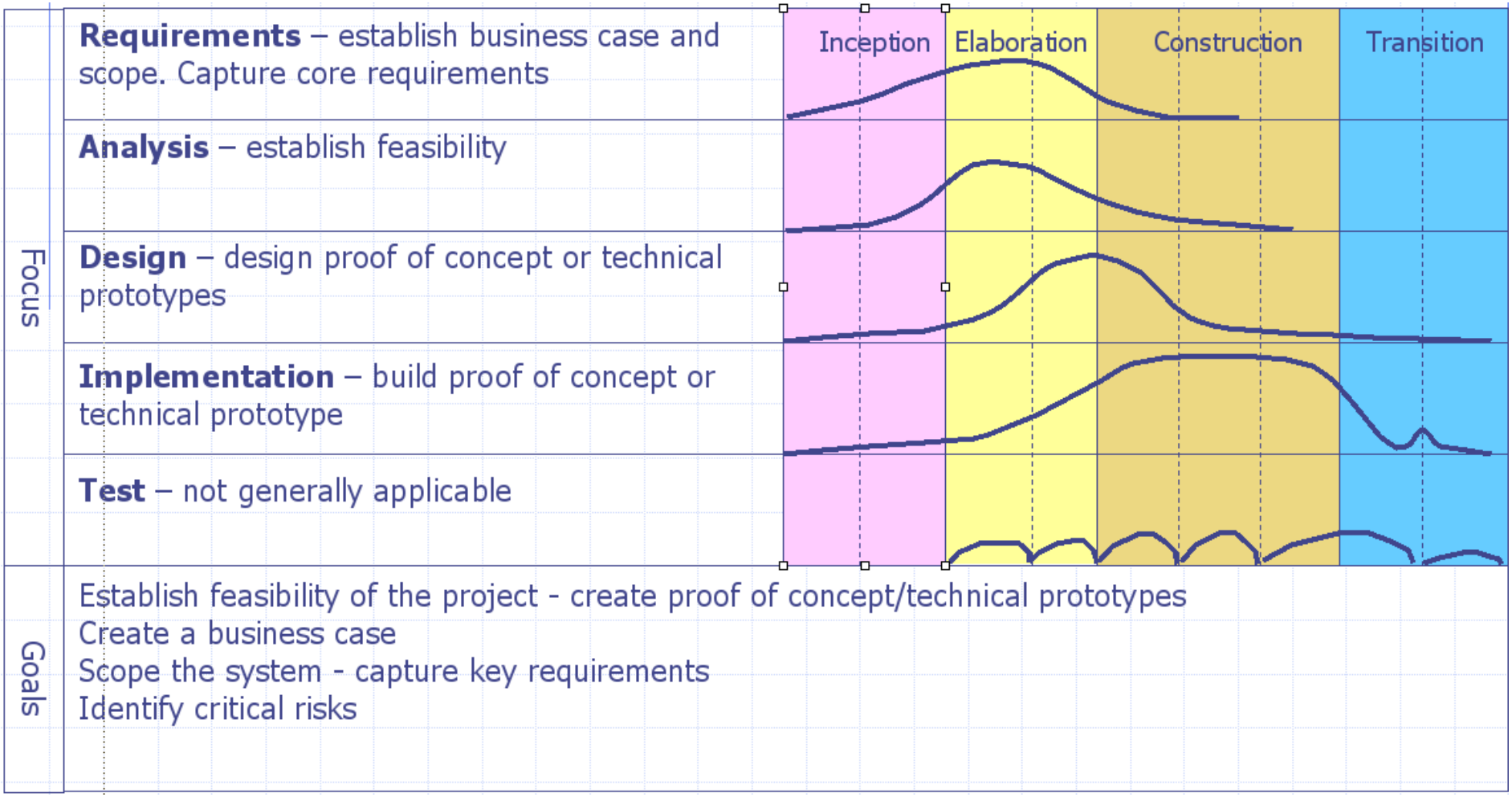
- Dipendentemente dalla dimensione del progetto ogni fase può includere diverse iterazioni per ognuna delle quattro fasi
- Una fase si conclude con una milestone
- Fasi caratterizzate da: obiettivi, focus dei flussi di lavoro (attività), milestone

Fasi e flussi di lavoro dello UP



Avvio

(Inception)



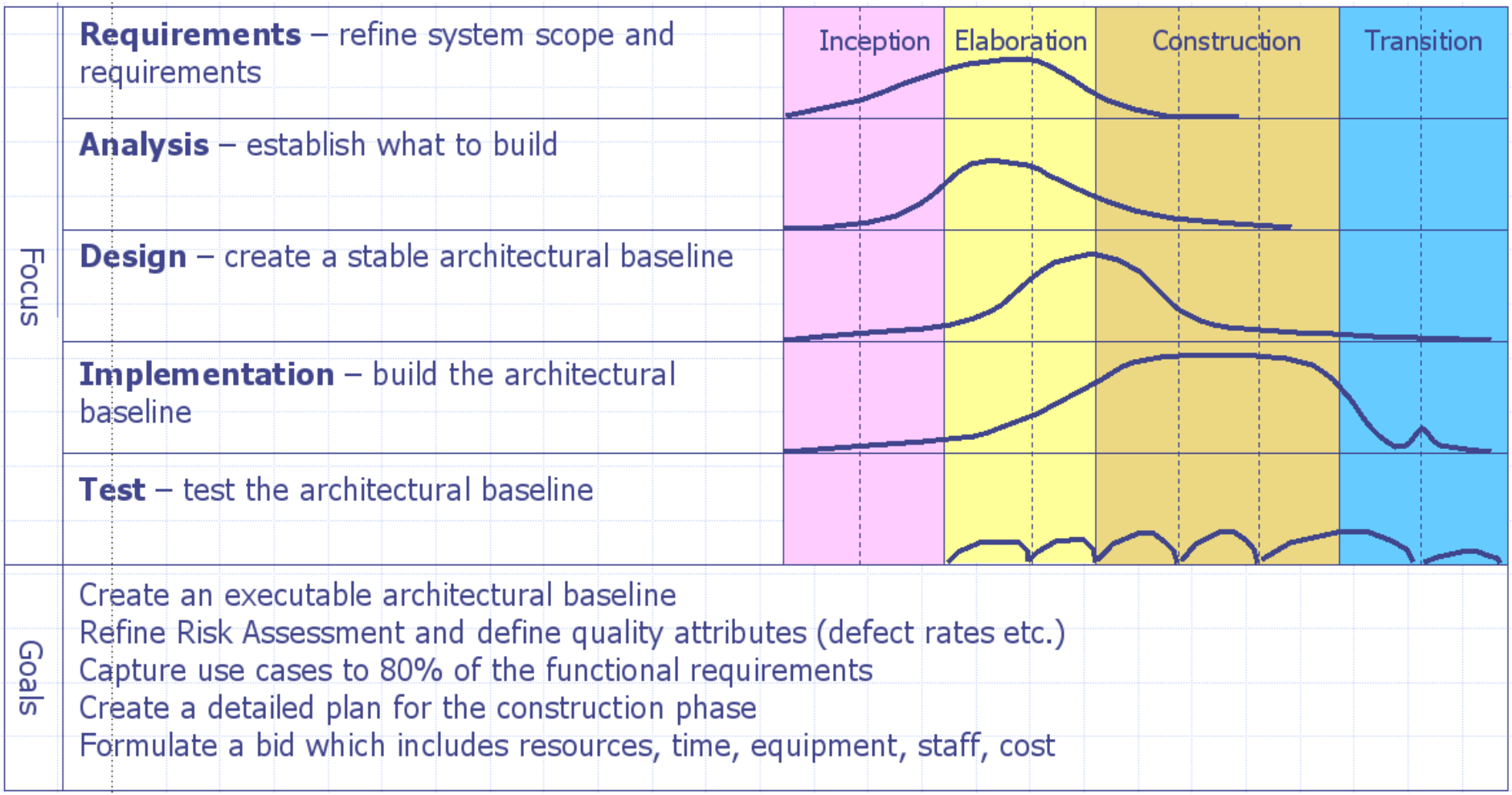
Avvio

(Inception)

- Condizioni da soddisfare dalla fase e artefatti attesi:
  - Lo scopo del sistema è stato definito (documento di alto livello con i requisiti principali)
  - Definire l'ambito del progetto (I requisiti base del sistema sono stati catturati e concordati con i vari attori – Casi d'uso 10-20 %)
  - Una "visione" architetturale è stata definita (documento iniziale di architettura)
  - Una prima valutazione del rishio è stata derivata (documento o archivio di valutazione dei rischi)
  - Una prima valutazione dei costi e dei tempi (pianificazione del progetto)
  - Un caso illustrativo di business è stato prodotto con evidenziazione dei vantaggi (caso di business)
  - Fattibilità del progetto è stata confermata (definizione di uno o più prototipi)

Elaborazione

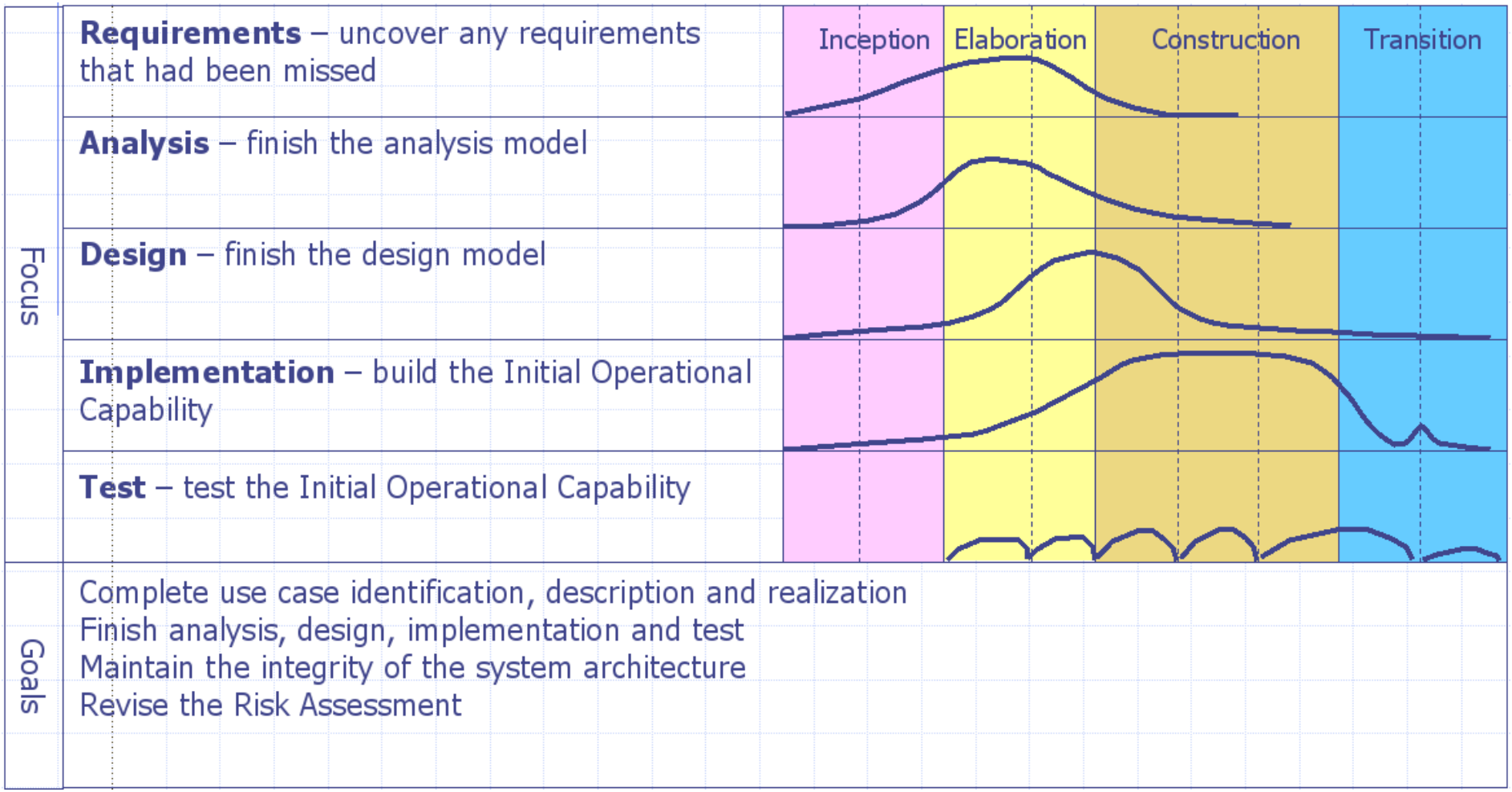
(Elaboration)



Elaborazione
(Elaboration)

- Condizioni da soddisfare dalla fase ed artefatti attesi:
  - Creare Baseline robusta ed eseguibile (la baseline è eseguibile)
  - Baseline eseguibile dimostra che i rischi sono stati identificati e risolti (modello statico UML, modello dinamico UML, casi d'uso)
  - Revisione della stima dei rischi (valutazione aggiornata)
  - Visione d'insieme del sistema stabilizzata (80 % dei casi d'uso definiti - documento descrittivo del sistema)
  - La parti hanno approvato la pianificazione di progetto
  - pianificazione a livello di dettaglio tale da consentire formulazioni realistiche di tempi, costi e risorse richieste (pianificazione aggiornata)
  - Accordo finale tra le parti (Accordo firmato)

Costruzione
(Construction)

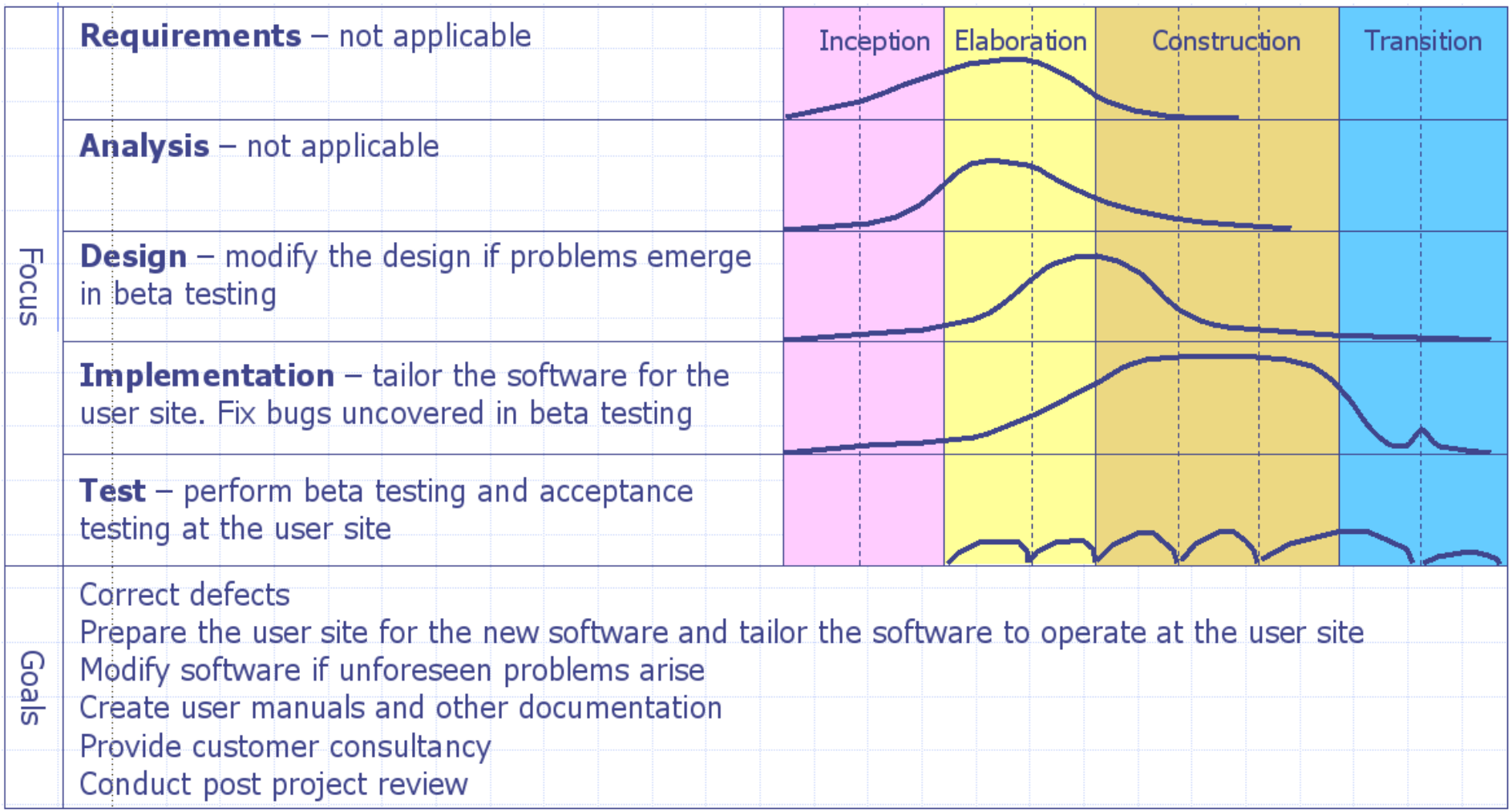


Costruzione
(Construction)

- Condizioni da soddisfare dalla fase ed artefatti attesi:
  - Software stabile e di qualità sufficiente ad essere rilasciato (prodotto software, modello UML, suite di test)
  - verifica dei costi rispetto alla pianificazione (pianificazione del progetto)
  - Accordo tra le parti per il rilascio (manuale utente e descrizione del rilascio)

Transizione

(Transition)



Transizione
(Transition)

- Condizioni da soddisfare dalla fase ed artefatti attesi:
  - Il beta test è ultimato, le necessarie modifiche al software sono state effettuate, gli utenti concordano con l'affermare che il sistema è stato rilasciato con successo (prodotto software)
  - Gli utenti utilizzano attivamente il prodotto (pianificazione del supporto)
  - definizione delle strategie di supporto (manuali utente aggiornati)