Università degli studi di Modena e Reggio Emilia

Dipartimento di Ingegneria "Enzo Ferrari"
Corso di Laurea Triennale in Ingegneria Informatica

Elaborato della prova finale nell'ambito dell'insegnamento di Ingegneria del Software

Photoshop Document Processor (PDP):

– software per l'automazione di elaborazioni di documenti su Adobe Photoshop –

Studente:	Marco Michelini
E-Mail:	marco.michelini@protonmail.com
Relatore:	Francesco Guerra

Indice

1 INTRODUZIONE....3
1.1 Descrizione del problema....3
1.2 Principi di progetto....4
1.3 Organizzazione dell'elaborato....5

2 REQUISITI E ASSUNZIONI....5
2.1 Requisiti....6
2.1.1 Sovrimpressione nome documento....7
2.1.2 Sovrimpressione codice numerico documento....8
2.1.3 Tonalizzazione documento....8
2.2 Assunzioni....10
2.3 Diagrammi di supporto....11

3 DESIGN....16
3.1 Architettura....16
3.2 Componenti di PDP....17
3.2.1 FiltroSovrimpressione....19
3.2.2 TabellaToniCMYK....19
3.2.3 FiltroTonalizzazioneCMYK....19

4 IMPLEMENTAZIONE....20
4.1 Convenzioni....20
4.2 Confronto prestazionale tra algoritmi di calcolo dei fattori di tonalizzazione….21
4.3 Definizione e invocazione di pipeline di esempio....24

5 CONCLUSIONI....28

6 APPENDICI....28

1 Introduzione

Con il presente elaborato, si vuole documentare e descrivere in maniera il più possibile dettagliata un software per l'automazione di elaborazioni di documenti, progettato e implementato dal sottoscritto per integrarsi con Adobe Photoshop ed estenderne le funzionalità.

Tale software, che sarà d'ora in avanti indicato con l'acronimo PDP (Photoshop Document Processor), vuole porsi sia come applicazione delle conoscenze acquisite nel linguaggio di programmazione JavaScript sia come soluzione ad un problema di natura pratica in ambito ceramico.

1.1 Descrizione del problema

Come accennato, PDP nasce per rispondere ad un'esigenza concreta: di conseguenza è di vitale importanza, prima di procedere, spiegare quale sia questa esigenza. E' altresì fondamentale descrivere quello che è l'ambiente in cui il software si troverà ad operare (dovendo questo mutare ed evolversi in risposta a sue variazioni per conservare la sua utilità).

Nella fattispecie, PDP vuole fornire una soluzione alla stringente necessità di automatizzare elaborazioni di documenti su Adobe Photoshop: all'interno di aziende ceramiche, è non solo comune ma essenziale far uso di software di grafica al fine di definire colori, motivi e grafiche delle piastrelle che si andranno poi a produrre. Questo richiede inevitabilmente un gran numero di elaborazioni su tali piastrelle (o, per essere più precisi, sui documenti che ne sono la rappresentazione digitale), elaborazioni che tendono ad avere un grado elevato di ripetitività. Si pensi, ad esempio, alla necessità di bilanciare il tono[1] delle piastrelle di uno stesso lotto in modo che queste appaiano uniformi all'occhio umano, riportando ciascuna di queste ad uno specificato riferimento. Per gestire simili ripetizioni, Photoshop rende disponibili due strumenti: le azioni (registrazioni di tutti gli input dell'utente, che possono essere invocate a piacimento una volta salvate), e gli script (che permettono di controllare ogni funzionalità alla quale l'utente accede tramite

[1] Tono: proprietà di un colore data da tonalità, saturazione e luminosità. Nel contesto del presente elaborato, si riferisce anche all'intero documento (in particolare, immagine), sottintendendo che tale quantità è valutata rispetto a valori di colore medi.

interfaccia grafica). I primi sono sufficienti quando l'elaborazione può essere ricondotta ad una serie di input sempre uguali, ma non in tutti quei casi in cui si richiedono input diversi a seconda del contesto o del documento. I secondi, sebbene completi, richiedono competenze di programmazione che esulano dal campo di competenze dell'utente medio.

In ultima analisi, in ambito ceramico non esiste un mezzo che sia al contempo completo e intuitivo per automatizzare le funzionalità offerte da Photoshop: ciò costringe inevitabilmente chi lavora in tale campo a perdere anche ore di tempo in compiti ripetitivi, con l'evidente calo di efficienza e produttività che ciò comporta. E' a questa mancanza che PDP intende sopperire.

1.2 Principi di progetto

Tenendo conto di quanto è stato detto nella sezione precedente, si possono identificare quelli che sono stati elevati a principi dell'intero progetto, e che lo hanno condizionato in ogni sua fase:

- Semplicità
  PDP deve essere intuitivo e conveniente per chiunque sia già abituato a lavorare con Photoshop in ambito ceramico.
- Manutenibilità
  Architettura e scelte di design devono privilegiare la manutenibilità, favorendo così l'aggiornamento e l'estensione di PDP nel tempo.
- Estensibilità
  Architettura e scelte di design devono tenere conto della concreta eventualità che nuove esigenze possano nascere nel tempo, richiedendo di estendere le funzionalità offerte da PDP.
- Autonomia
  PDP deve limitare il più possibile l'intervento dell'utente, richiedendolo solo quando strettamente necessario.

- Configurabilità

  PDP deve, nel modo più trasparente e conveniente possibile, dare la possibilità all'utente di configurare il suo funzionamento secondo le proprie preferenze.

1.3 Organizzazione dell'elaborato

Si conclude questo capitolo introduttivo con un breve riepilogo dei capitoli che seguono e dei loro contenuti:

- Capitolo 2: Requisiti e assunzioni

  Descrive nel dettaglio i requisiti che a PDP è richiesto di soddisfare, con particolare riguardo ai requisiti funzionali, riporta assunzioni degne di nota.
- Capitolo 3: Design

  Riporta scelte e principi di design, descrive l'architettura adottata e i suoi componenti principali.
- Capitolo 4: Implementazione

  Definisce le convenzioni e le scelte impiegate nell'implementazione e presenta un esempio d'uso di PDP.
- Capitolo 5: Conclusioni

  Riporta considerazioni sul progetto descritto dal presente elaborato e sulla sua implementazione.
- Capitolo 6: Appendici

  Elenca alcune letture di approfondimento, d'interesse per meglio comprendere come PDP si integri con Photoshop e ne invochi le funzionalità.

2 Requisiti e assunzioni

Nel presente capitolo si riportano i requisiti che PDP è chiamato a soddisfare (assieme ai relativi diagrammi descrittivi, ove ritenuto particolarmente utile) e le assunzioni che sono state fatte relativamente ad essi. Requisiti e assunzioni sono stati valutati e concordati con la partecipazione di un esperto che lavora in un'azienda ceramica, nell'esatto contesto delineato nel primo capitolo.

2.1 Requisiti

Oltre a rispettare i principi di progetto già esposti, PDP deve:

- Essere invocato mediante script di Photoshop.
- Permettere elaborazioni in sequenza sullo stesso set di documenti.
- Una volta invocato, elaborare solamente i documenti già aperti in Photoshop.
- Lasciare invariate le preferenze[2] di Photoshop.
- Essere compatibile con Adobe Photoshop CS5 e versioni successive.
- Essere compatibile con Windows 10, Windows 10 Pro e versioni successive.
- Essere compatibile con macOS Catalina e versioni successive.
- Garantire prestazioni tali da determinare un risparmio di tempo rispetto allo svolgimento manuale delle elaborazioni implementate. Si prende come riferimento il caso in cui tali elaborazioni siano svolte dall'utente medio.

Con riferimento ai requisiti di cui sopra, si riporta un semplice use case diagram a rappresentazione del contesto di utilizzo tipico:

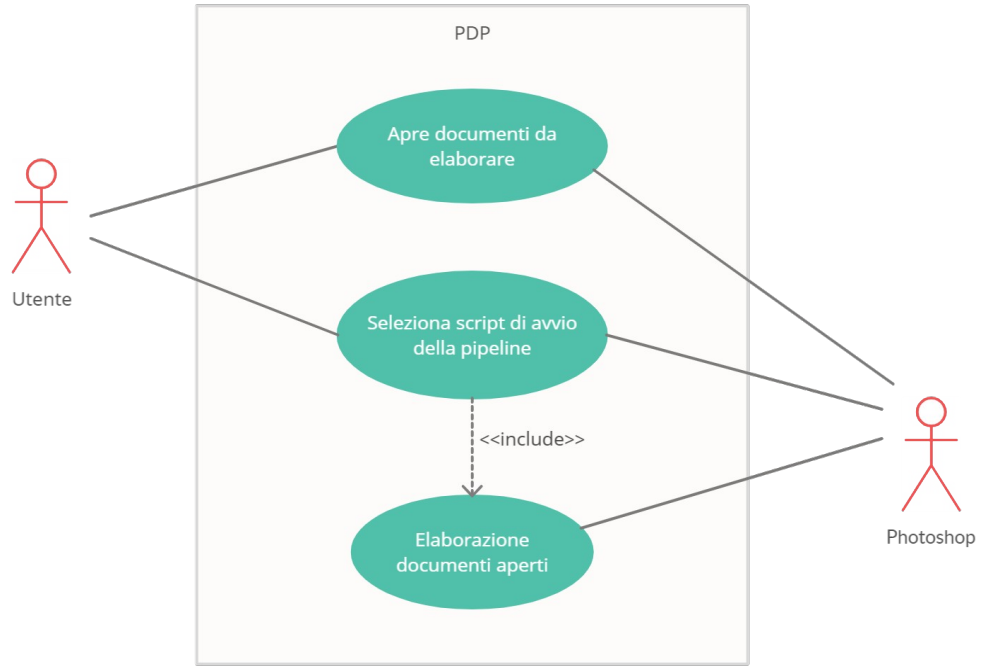

Figura 1: Use Case Diagram - Contesto di utilizzo tipico.

[2] Preferenze: impostazioni di Photoshop, quali l'unità di misura dei righelli, impostabili dall'omonima finestra di dialogo.

Si noti come PDP agisca da ponte di collegamento tra l'utente e Photoshop, non facendo altro che automatizzare funzionalità offerte da quest'ultimo (estendendole laddove necessario).

Si richiede altresì di implementare le seguenti elaborazioni:

2.1.1 Sovrimpressione nome documento

Per ogni documento aperto, si richiede di sovrimprimere su di esso il suo nome, privato del formato (si ricorda che PDP sarà invocato sia in ambienti Windows che macOS). Le caratteristiche del testo corrispondente (font, colore, dimensione, ecc.) devono essere fornite mediante un'azione di configurazione resa disponibile dall'utente. Tale azione deve creare un nuovo livello[3] e inserirvi un campo di testo con le caratteristiche e nella posizione approssimativa volute. Il nome del documento deve dunque andare a sostituire il contenuto di tale campo, il quale dovrà poi essere riposizionato in base alla sua posizione approssimativa (se il campo di testo era all'incirca centrato dovrà essere centrato in maniera esatta, se era spostato verso il bordo sinistro/destro del documento dovrà essere allineato a tale bordo). Una volta riposizionato, il campo di testo deve lasciare un margine di 4 cm dal bordo superiore del documento e da quello a cui è stato eventualmente allineato. Evidentemente, si deve fare in modo di evitare, per quanto possibile, che il campo di testo fuoriesca dai bordi del documento. Terminate tali operazioni, tutti i livelli di ogni documento devono essere uniti[4] e i documenti devono essere salvati.

Si rimanda alla sezione Assunzioni per i dettagli relativi al formato in cui i file saranno nominati, che in ogni caso sarà sempre lo stesso per tutti i documenti rappresentanti piastrelle dello stesso lotto.

[3] Livello: in Photoshop, rappresenta un singolo strato di un documento. Un documento si compone di uno o più livelli, tra loro indipendenti: qualsiasi modifica fatta su di un livello ha effetto solamente su di esso. Ogni documento è dunque ottenuto impilando i livelli di cui è composto uno sull'altro.

[4] In Photoshop, più livelli possono essere uniti in un unico livello. Si noti che, una volta salvato e chiuso un documento i cui livelli sono stati uniti, non è più possibile risalire ai livelli originari.

2.1.2 Sovrimpressione codice numerico documento

L'elaborazione è sostanzialmente identica alla sovrimpressione del nome di un documento, con l'unica differenza data dal testo che si vuole sovrimprimere: questo sarà costituito infatti dal codice numerico del documento, dato da un numero intero che identifica la piastrella corrispondente fra quelle del lotto cui appartiene.

Si rimanda alla sezione Assunzioni per i dettagli relativi al formato in cui i file saranno nominati.

2.1.3 Tonalizzazione documento

Per meglio chiarire questa particolare elaborazione, prima di tutto si espone la procedura seguita per andare a tonalizzare[5] una serie di documenti su Photoshop, procedura che si articola nelle fasi seguenti:

i. Rilevazione del tono dei documenti
   Per ciascun documento, richiede:

   (a) La duplicazione del livello di sfondo del documento.
   (b) L'applicazione del filtro sfocatura media[6] di Photoshop al duplicato del livello di sfondo.
   (c) La rilevazione del tono del documento, che consiste in pratica nella lettura delle quantità percentuali di ciascun canale[7].

ii. Scelta del tono di riferimento
   Esistono diversi approcci per operare tale scelta. Principalmente, si segue uno degli approcci riportati di seguito:

[5] Nel presente elaborato, s'intende la tonalizzazione come l'elaborazione con cui il tono di un documento (in particolare, di un'immagine) viene portato a coincidere con un tono di riferimento.

[6] Filtro sfocatura media: filtro di Photoshop che calcola la media delle quantità percentuali di ogni canale, valutandola su tutti i pixel del livello al quale è applicato, per poi riempire il livello con una tinta piatta (ovvero, un colore omogeneo) identificata dalla n-pla delle medie calcolate.

[7] Canale: in Photoshop, ogni documento è dato dalla composizione di un certo numero di canali, costituiti da immagini in scala di grigio. Il numero di questi canali dipende dal metodo colore scelto per il documento, che a sua volta determina il modo in cui i colori vengono rappresentati. Nel corso del presente elaborato, si farà sempre riferimento al metodo colore CMYK, che esprime ogni colore in termini di quantità percentuali dei quattro canali Ciano, Magenta, Giallo e Nero.

(a) Si prende a riferimento il tono di uno dei documenti.

(b) Si prendono a riferimento i toni di più documenti. Ciò viene fatto andando a definire per ogni canale, in base ai documenti scelti, un range di quantità percentuali accettabili. I canali di un documento la cui quantità percentuale rientra nel range non subiranno variazioni.

iii. Bilanciamento del tono dei documenti

Da applicare ad ogni documento, richiede di aggiungervi un miscelatore canale[8] e, per ogni canale, trovare la quantità percentuale di canale da aggiungere per portarlo a riferimento. Il tono del documento è dunque a riferimento quando sono a riferimento le quantità percentuali dei suoi canali. Le dette quantità percentuali saranno chiamate da qui in poi "fattori di tonalizzazione" per brevità.

iv. Eliminazione dei duplicati dei livelli di sfondo dai documenti

Si eliminano i livelli duplicati, avendo questi l'unica funzione di permettere la rilevazione del tono.

v. Unione dei livelli di ciascun documento

vi. Salvataggio dei documenti

Fatta questa premessa, si richiede di effettuare la tonalizzazione di tutti i documenti aperti, a partire da un riferimento fornito dall'utente. In particolare, PDP deve: misurare il tono di tutti i documenti aperti, compilare, visualizzare e scrivere su file di testo una tabella dei toni che riporti tali letture, visualizzare e scrivere su file di testo la quantità percentuale media di ogni canale (valutata su tutti i documenti), richiedere all'utente un valore o un range di riferimento (la cui validità deve essere verificata) per ciascun canale, bilanciare il tono dei documenti aperti, eliminare i livelli duplicati, unire tutti i livelli di ciascun documento e infine salvare tutti i documenti.

Relativamente a tale procedura, si fanno alcune importanti precisazioni:

[8] Miscelatore canale: in Photoshop, è uno strumento che consente di aumentare o ridurre le quantità percentuali dei canali di un documento, determinandone in tal modo una variazione di tono. Si noti che questo è possibile fino ad un incremento massimo del +200%, oltre il quale è necessario aggiungere ulteriori miscelatori canale.

- Il valore o range di riferimento inserito dall'utente per un canale viene utilizzato, per determinare la quantità percentuale alla quale il canale deve essere portato con la tonalizzazione, nel modo seguente: se il riferimento inserito è un valore, il canale dovrà essere portato a tale valore; se il riferimento inserito è un range del tipo [*min*, *max*], il canale dovrà essere portato a *min* se la sua quantità percentuale è inferiore a *min*, a *max* se la sua quantità percentuale è superiore a *max*, e dovrà essere lasciato invariato se la sua quantità percentuale cade nel range.
- La tabella dei toni deve avere tante righe quanti sono i documenti (ciascuna identificata dal codice numerico del documento corrispondente) e tante colonne quanti sono i canali.
- Le quantità percentuali di canale devono essere arrotondate all'intero più vicino.

2.2 Assunzioni

Nella presente sezione si riportano tutte le assunzioni relative alle tre elaborazioni appena descritte. Queste sono giustificate dalle procedure dell'azienda ceramica che si è presa come riferimento, le quali possono considerarsi relativamente stabili. Tuttavia, come si avrà modo di vedere nel capitolo seguente, il design di PDP è sufficientemente flessibile da poter essere adattato a variazioni che dovessero in futuro rendere invalide tali assunzioni.

Con riferimento alla sovrimpressione del nome e del codice numerico:

- Si assume che il nome dei documenti da processare sia nel formato CN_STR.FFF, dove CN è il codice numerico (un intero a due cifre da 00 a 99), STR è una stringa che utilizza il carattere '_' come separatore di parola e infine FFF è l'estensione del documento. Tale formato sarà detto "standard" per comodità.
- Si assume che il nome dei documenti da processare e/o la dimensione del font usato per il testo dell'azione di configurazione non siano tali da far fuoriuscire il testo sovrimpresso dai bordi del documento.

Con riferimento alla tonalizzazione:

- Si assume che ogni documento sia tonalizzabile usando un singolo miscelatore canale, ovvero che non sia mai necessario applicare fattori di tonalizzazione superiori al +200%.
- Si assume che ogni documento utilizzi il metodo colore CMYK[9].
- Si assume che tra i documenti da elaborare non ci sia alcun documento con uno o più canali aventi una quantità percentuale pari allo 0%.[10]

2.3 Diagrammi di supporto

Si conclude il secondo capitolo riportando una serie di activity diagram volti a rappresentare visivamente le tre elaborazioni che PDP deve implementare. Si noti l'assenza di un diagramma per la sovrimpressione del codice numerico di un documento: questo sarebbe stato fondamentalmente identico a quello per la sovrimpressione del nome di un documento, e pertanto si è deciso di ometterlo.

Inoltre, si osserva che tali diagrammi sono da considerarsi congiuntamente ai requisiti e alle assunzioni esposti, essendo stati pensati non tanto per catturare in maniera completa le elaborazioni da implementare, quanto piuttosto per darne una visione d'insieme di più pratica consultazione.

[9] Il metodo colore CMYK esprime ogni colore in termini di quantità percentuali di quattro canali: Ciano, Magenta, Giallo e Nero. Attualmente rappresenta il metodo colore più usato dall'azienda ceramica di riferimento.

[10] In pratica ciò accade raramente, e quando accade richiede una procedura diversa (in particolare, l'introduzione di canali di supporto) per procedere con la tonalizzazione. Per questo motivo, e per la scarsità di tempo a disposizione, si è scelto di demandare la gestione di questa possibilità a versioni future di PDP.

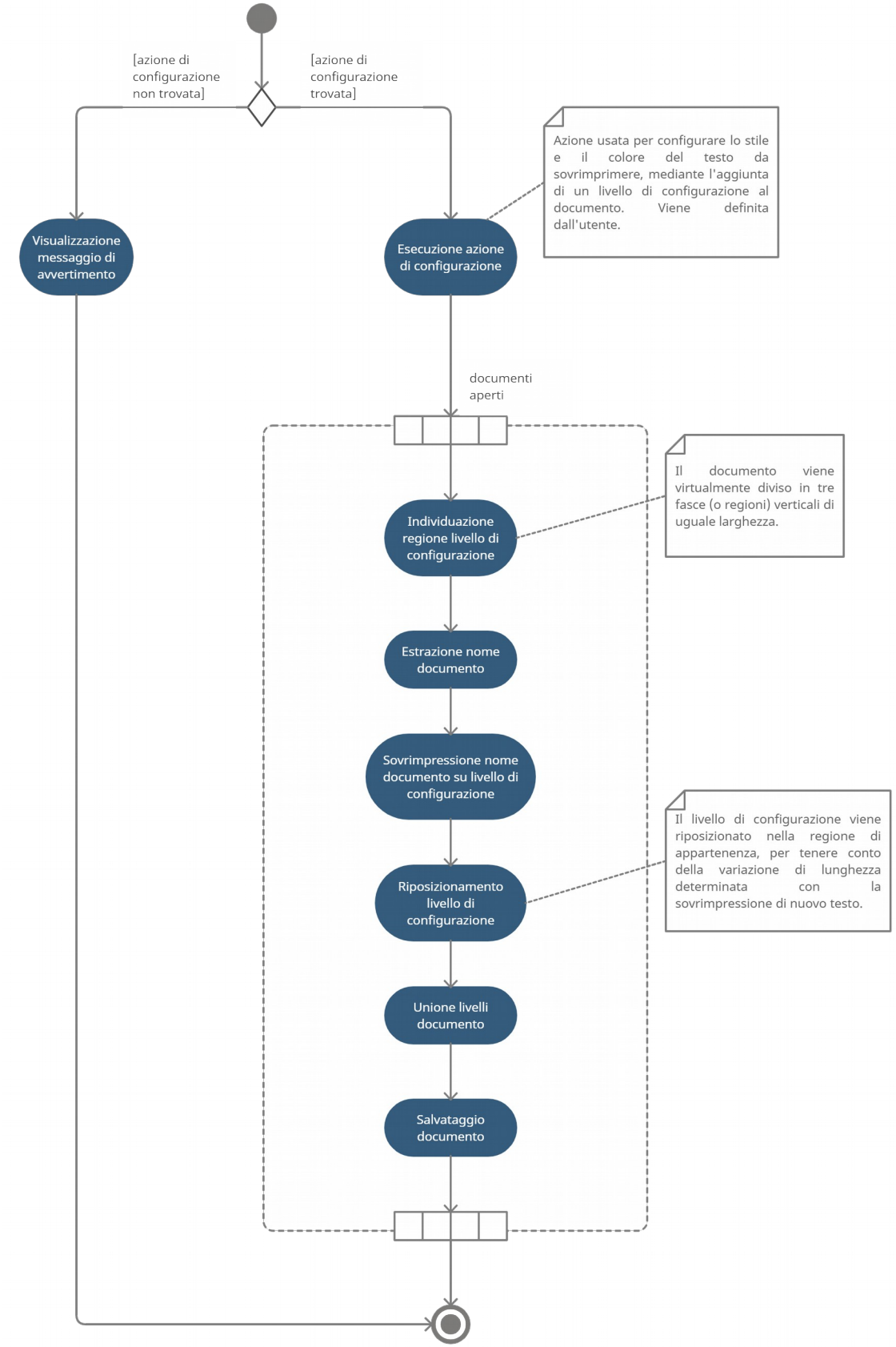

Figura 2: Activity Diagram - Sovrimpressione nome documento.

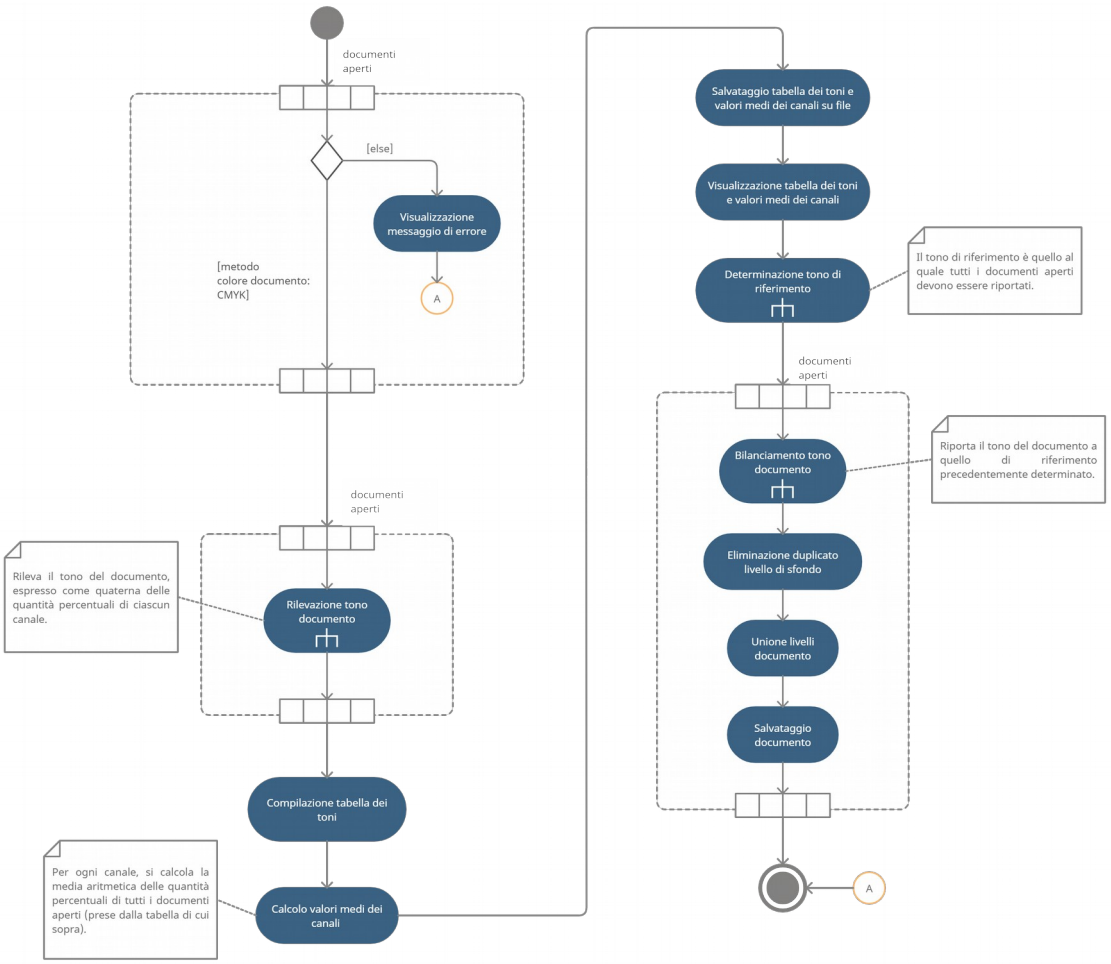

Figura 3: Activity Diagram - Tonalizzazione documento.

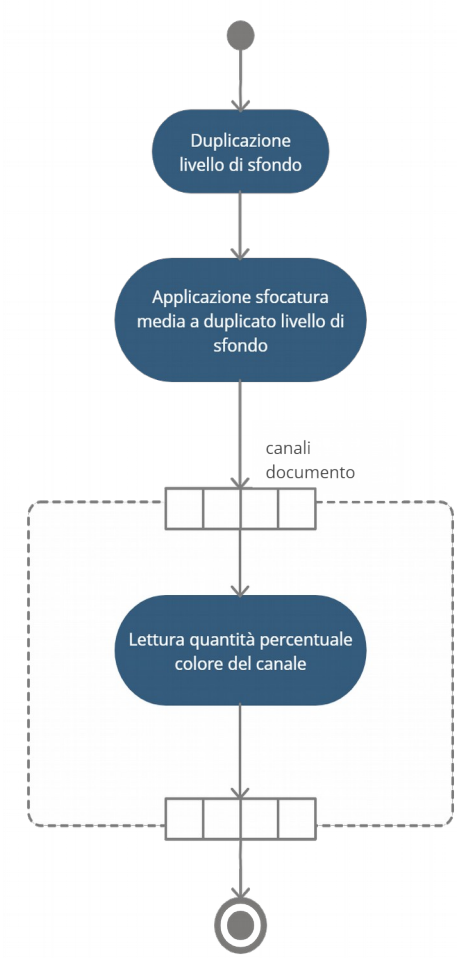

Figura 4: Activity Diagram - Rilevazione tono documento.

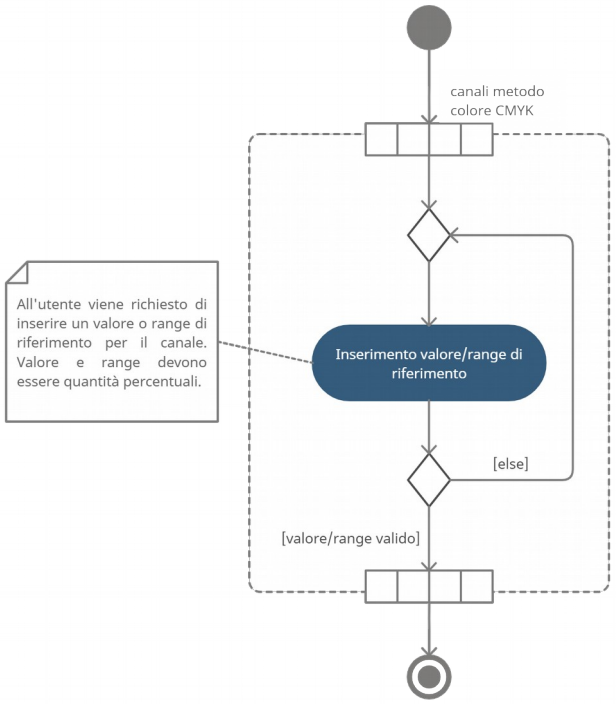

Figura 5: Activity Diagram - Determinazione tono di riferimento.

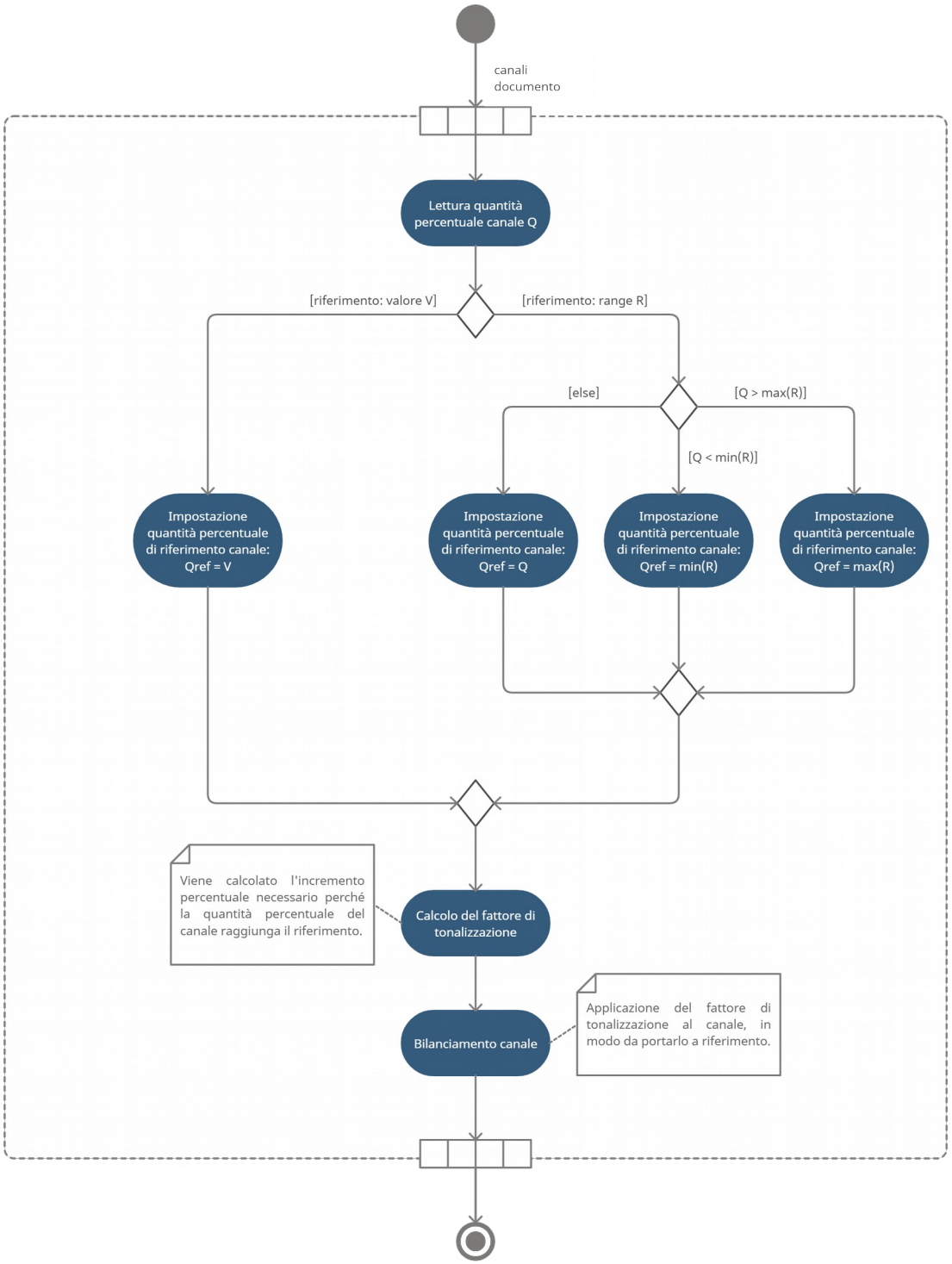

Figura 6: Activity Diagram - Bilanciamento tono documento.

3 Design

In questo e nel successivo capitolo, si vedrà come i principi dell'Ingegneria del Software, i principi di progetto e, in particolar modo, le nuove conoscenze acquisite nel corso di tale progetto (nella programmazione in linguaggio JavaScript, ma anche nell'uso di Photoshop, in ambito ceramico) siano stati applicati nel design e nell'implementazione di PDP.

Si descriveranno dunque l'architettura del software e i suoi componenti principali, le convenzioni seguite, le scelte implementative fatte. Si mostrerà inoltre il funzionamento di PDP, tramite l'analisi di un caso pratico.

3.1 Architettura

Considerando tutto quanto detto fino ad ora, si può osservare come il problema fondamentale che PDP deve risolvere sia l'implementazione di elaborazioni, sostanzialmente di tipo batch, che possano essere messe in sequenza una dopo l'altra e che processino lo stesso set di documenti. Si consideri anche che set di documenti diversi possono richiedere diverse elaborazioni, e che nel tempo nascerà certamente l'esigenza di nuove e diverse operazioni, da concatenare in modi altrettanto variegati.

L'architettura di PDP deve supportare tutte queste esigenze, oltre ai principi già esposti nel primo capitolo. A tale scopo, si è scelto di adottare un'architettura di tipo pipe-and-filter, in cui i filtri (che corrispondono alle diverse elaborazioni da implementare), eventualmente configurati mediante apposite azioni fornite dall'utente, vengono concatenati uno dopo l'altro a formare delle pipeline. Ogni pipeline opera dunque su di un certo set di documenti, ed invoca su di essi i filtri che la compongono, nell'ordine in cui questi sono stati aggiunti.

In questo contesto, è doveroso osservare come l'anello debole sia costituito dall'interfaccia tra due filtri: non potendo prevedere come e quali filtri vengano combinati a runtime, è essenziale assicurarsi che tali interfacce siano standardizzate, in modo da poter evitare situazioni anomale che possano innescare criticità nel software.

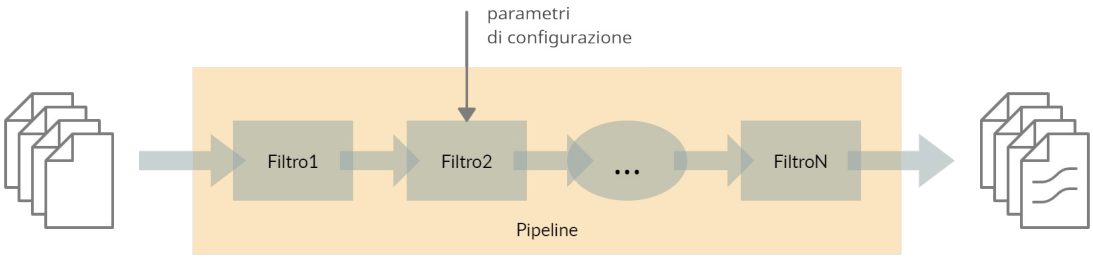

Figura 7: Pipeline di PDP - Una pipeline processa dei documenti mediante gli N filtri di cui è costituita, ciascuno dei quali implementa una diversa elaborazione e può ricevere o meno dei parametri di configurazione dall'utente.

3.2 Componenti di PDP

Nella presente sezione si riportano in un class diagram e si descrivono i componenti di PDP più rilevanti per l'implementazione dell'architettura e delle funzionalità esposte precedentemente.

Come premessa, si osservi che nel diagramma è possibile identificare due astrazioni fondamentali: Pipeline e Filtri. Le loro istanze svolgono le proprie funzioni collaborando con specifici componenti di supporto, eventualmente ricevendo dei parametri dall'utente mediante delle azioni di configurazione.

Si osservi anche come PDP si integri con astrazioni fornite da Photoshop (Document, ArtLayer, SolidColor, ColorSampler) per implementare le funzionalità richieste. Per dettagli relativamente ad esse si rimanda alle Appendici, fornendone qui solamente una descrizione sommaria:

- Document

  Rappresenta un singolo documento su Photoshop, incapsulandone tutte le caratteristiche (come i livelli che lo compongono).
- ArtLayer

  Rappresenta un singolo livello di un documento. I livelli di un documento sono accessibili dalla proprietà artLayers di Document, in cui sono memorizzati sotto forma di array.

- SolidColor

  Rappresenta un singolo colore in termini di quantità percentuali dei canali di uno specifico metodo colore. Nello specifico, offre diverse rappresentazioni dello stesso colore, una per ogni metodo colore supportato da Photoshop. Usato da PDP per rappresentare il tono di un documento.

- ColorSampler

  Oggetto che permette di campionare il colore di uno specifico livello in un documento, rappresentato come SolidColor.

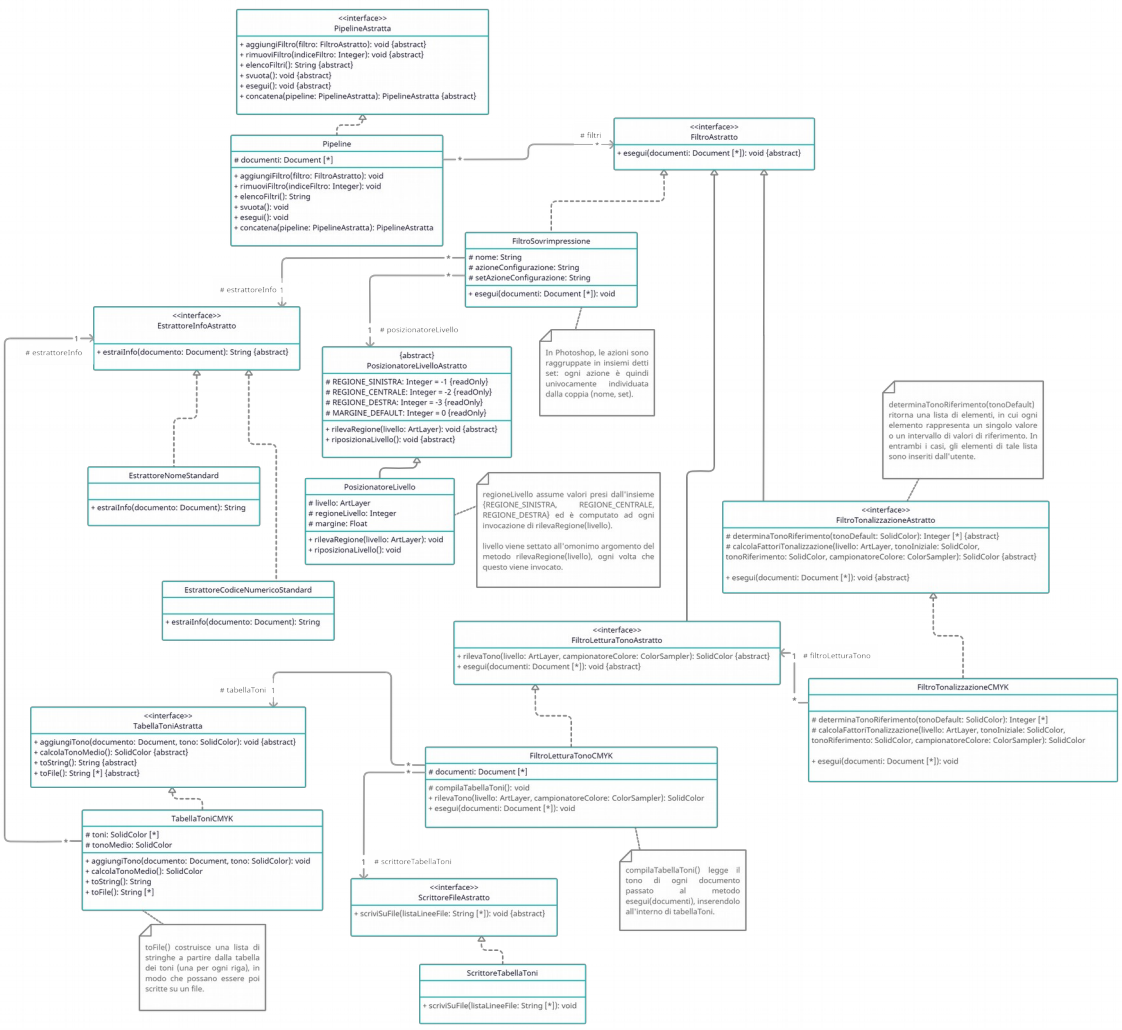

Figura 8: Componenti di PDP. In caso di informazioni mancanti, si deve far riferimento ai relativi valori di default, a eccezione delle associazioni (da considerarsi sempre unidirezionali).

3.2.1 FiltroSovrimpressione

Gestisce la sovrimpressione di testo su di un documento, configurandosi mediante un'azione settata dall'utente. Nello specifico, il testo sovrimpresso viene estratto dai metadati del documento. L'estrazione delle informazioni del documento e il riposizionamento del campo di testo (dopo la sovrimpressione) sono demandate a componenti esterni, rispettivamente EstrattoreInfoAstratto e PosizionatoreLivelloAstratto. Lo specifico tipo di EstrattoreInfoAstratto può essere settato a run-time, di modo che FiltroSovrimpressione può sovrimprimere sia il nome che il codice numerico di un documento, a seconda della situazione.

3.2.2 TabellaToniCMYK

Gestisce la creazione e la formattazione testuale di una tabella dei toni, sia per la sua visualizzazione mediante finestra di dialogo, sia per il suo salvataggio su file da parte di ScrittoreTabellaToni. Fornisce anche un metodo per calcolare il tono medio, valutato su tutti i documenti. All'interno della tabella, ogni tono è individuato dal codice numerico del documento cui si riferisce. Usato da FiltroLetturaTonoCMYK.

3.2.3 FiltroTonalizzazioneCMYK

Gestisce la richiesta dei valori/range di riferimento per la tonalizzazione all'utente, la determinazione, a partire da essi, delle percentuali di riferimento di ciascun canale (come descritto nell'activity diagram "Bilanciamento tono documento") e infine la tonalizzazione vera e propria dei documenti. Si compone con FiltroLetturaTonoCMYK, a cui demanda la rilevazione del tono dei documenti e la loro organizzazione nella tabella dei toni. Notare che FiltroLetturaTonoCMYK è un filtro a tutti gli effetti, e quindi può essere utilizzato da solo laddove fosse necessario rilevare, ma non correggere, il tono di un set di documenti.

4 Implementazione

La prima e principale scelta implementativa è stata quella relativa al linguaggio da utilizzare per implementare PDP: come già anticipato si è scelto JavaScript, o più propriamente ExtendScript. ExtendScript è in pratica un'estensione di JavaScript conforme allo standard ECMAScript 3, al quale aggiunge una moltitudine di oggetti e funzionalità pensati per facilitare lo scripting per applicazioni Adobe.

I motivi alla base di questa scelta sono stati tre:

- Rispetto agli altri linguaggi supportati da Photoshop (AppleScript, VBScript, Visual Basic), ExtendScript è l'unico che possa essere usato sia su sistemi Windows che su sistemi macOS.
- ExtendScript è fondamentalmente identico a JavaScript.
- JavaScript è molto utilizzato nello sviluppo di applicazioni web, pertanto scegliere ExtendScript ha dato anche modo al sottoscritto di imparare un nuovo linguaggio che, sicuramente, si rivelerà nuovamente utile in futuro.

Questa scelta ha avuto diverse conseguenze sull'implementazione di PDP, come si vedrà in seguito.

4.1 Convenzioni

Basandosi su una versione alquanto datata di ECMAScript, ExtendScript prevede diverse limitazioni di cui si è dovuto tener conto per implementare PDP. Seguono alcuni esempi particolarmente rilevanti:

- ExtendScript non definisce il concetto di classe: per creare più oggetti[11] a partire dallo stesso "modello", è necessario utilizzare una funzione costruttore[12].

[11] In ExtendScript, un oggetto è un tipo di dato strutturato che racchiude dati eterogenei (detti attributi) e funzioni (dette metodi), sotto forma di coppie "chiave: valore".

[12] Analoga al costruttore di una classe in linguaggi come Java, con la differenza che viene definita esternamente come semplice funzione, piuttosto che come metodo speciale della classe.

- ExtendScript non definisce il concetto di visibilità di attributi/metodi: non esiste alcun costrutto per limitare l'accesso ad attributi e metodi di un oggetto, che quindi sono sempre accessibili ovunque[13].
- ExtendScript manca di diversi oggetti e funzioni di uso comune, introdotti con versioni successive di ECMAScript.

Di conseguenza, si è deciso di fissare le seguenti convenzioni:

- Le classi concrete sono implementate come funzioni costruttore.
- Le classi astratte e le interfacce sono implementate come oggetti, sono identificate dal suffisso "Astratto/a" e devono essere trattate allo stesso modo delle classi astratte e delle interfacce di linguaggi come Java.
- I metodi astratti non devono avere un'implementazione, ma solamente lanciare un errore a run-time se invocati.
- I nomi di attributi e metodi che devono essere considerati *protected* e *private* devono essere sempre preceduti dai caratteri "_" e "#" (rispettivamente), ed essere trattati allo stesso modo degli attributi e dei metodi *protected* e *private* di linguaggi come Java.

4.2 Confronto prestazionale tra algoritmi di calcolo dei fattori di tonalizzazione

Nell'implementazione, un altro aspetto di fondamentale importanza è stato la scelta dello specifico algoritmo con cui i fattori di tonalizzazione devono essere calcolati: la tonalizzazione è per sua natura un'operazione inefficiente, richiedendo un numero molto elevato di letture e aggiustamenti di tono, con effetti importanti sui tempi di esecuzione.

Per dare un'idea del possibile ordine di grandezza, si supponga di voler portare a riferimento un documento per cui si abbiano fattori di tonalizzazione, da determinare, dati dalla quaterna (200, 200, 200, 200)[14]. Un possibile approccio può essere quello di incrementare gradualmente ciascuno di questi fattori, applicarli mediante un miscelatore ca-

[13] Facendo riferimento a linguaggi come Java, si potrebbe anche dire che in ExtendScript tutti gli attributi e i metodi di un oggetto sono sempre *public*.

[14] Nella pratica, le piastrelle di uno stesso lotto non si discostano dal riferimento in maniera importante, per cui questa assunzione è irrealistica. Ciononostante, è utile per fare considerazioni qualitative sui tempi di esecuzione teorici.

nale e infine leggere la variazione di tono conseguente. Poiché questo deve essere fatto per ciascun canale, supponendo di fissare la quaterna di fattori iniziali (100, 100, 100, 100) il miscelatore dovrebbe essere applicato 400 volte, prima di raggiungere i fattori di tonalizzazione cercati. Il numero di letture di tono sarebbe anch'esso pari a 400. Considerando di voler tonalizzare 30 documenti[15] con i medesimi fattori di tonalizzazione, il numero di applicazioni del miscelatore salirebbe a 12000, così come quello delle letture di tono. Ne consegue che è indispensabile porsi problemi relativamente alle prestazioni di PDP nella tonalizzazione dei documenti e, di conseguenza, nella determinazione dei fattori di tonalizzazione.

In particolare, si sono considerati tre diversi algoritmi per il calcolo dei fattori di tonalizzazione:

i. cft_v1: <u>stima + incrementi e decrementi</u>
   Si calcola una stima dei fattori di tonalizzazione, con cui si inizializza l'algoritmo di incrementi e decrementi. L'idea è quella di ridurre il più possibile il numero di applicazioni del miscelatore e di letture di tono.

   Per il calcolo della stima, si usa la seguente formula[16]:

   $$ft = \frac{Qref}{Q} \cdot 100$$

   dove, per un generico canale:

   (a) ft indica la stima del fattore di tonalizzazione.[17]
   (b) Qref indica la quantità percentuale di riferimento per il canale.
   (c) Q indica la quantità percentuale iniziale del canale.

[15] E' in effetti abbastanza comune dover lavorare con 20-30 documenti per volta nell'azienda ceramica di riferimento.

[16] Questa formula non ha validità generale, ed è per questo che si usa per stimare (e non calcolare) i fattori di tonalizzazione. Tuttavia, si è visto che nella pratica conduce a stime che sono molto vicine alle quantità finali.

[17] Questa dovrà essere arrotondata all'intero più vicino, dato che le quantità percentuali di canale e i fattori di tonalizzazione sono visualizzati da Photoshop come numeri interi.

ii. cft_v2: incrementi e decrementi

Per ogni canale, si inizializza il fattore di tonalizzazione corrispondente al 100%, per poi incrementare (se Q < Qref) o decrementare (se Q > Qref) dell'1% tale fattore e applicarlo al documento finché la quantità percentuale del canale non raggiunge la quantità percentuale di riferimento.

iii. cft_v3: ricerca binaria

Per ogni canale, si applica un algoritmo di ricerca binaria sull'intervallo [0, 200] dei possibili fattori di tonalizzazione, fino a trovare il valore che, se applicato, porta il canale a riferimento.

Ciascuno di questi algoritmi è stato implementato e successivamente testato in due diverse condizioni, al fine di valutarne e confrontarne i tempi di esecuzione: il caso cosiddetto medio, in cui i documenti da tonalizzare non si discostano troppo dal tono di riferimento, e il caso cosiddetto peggiore, in cui per ogni documento si hanno fattori di tonalizzazione (da determinare) dati dalla quaterna (200, 200, 200, 200).
In entrambi i casi, sono stati valutati i tempi di esecuzione medi su quattro iterazioni e sugli stessi venti documenti (chiaramente differenti nel tono).

Si riporta di seguito il risultato di queste valutazioni:

	tempo di esecuzione medio [min]		tempo di esecuzione medio per documento [min/doc]	
	caso medio	caso peggiore	caso medio	caso peggiore
cft_v1	1,88	3,03	0,0940	0,1515
cft_v2	8,83	35,99	0,4415	1,7995
cft_v3	2,82	3,81	0,1410	0,1905

Tabella 1: Confronto dei tempi di esecuzione medi tra algoritmi di calcolo dei fattori di tonalizzazione, valutati su venti documenti.

L'algoritmo cft_v1 si è rivelato essere il più efficiente in entrambi i casi di analisi, ed è dunque quello utilizzato da PDP. Si osservi come per tutti e tre gli algoritmi il tempo di esecuzione medio sia aumentato nel passaggio da caso medio a caso peggiore, in particolare per quanto riguarda l'algoritmo cft_v2. A titolo di confronto, si precisa che per

eseguire l'intero processo di tonalizzazione sullo stesso numero di documenti, i quali ricadono in pratica sempre nel caso medio, l'utente tipico impiega circa 30 minuti.

4.3 Definizione e invocazione di pipeline di esempio

Si conclude il capitolo mostrando come i componenti precedentemente descritti collaborino per implementare le funzionalità richieste, prendendo a riferimento un caso concreto. Nello specifico, si consideri il caso in cui, dato un certo set di documenti, si voglia sovrimprimere su ciascuno di essi il proprio codice numerico e, successivamente, tonalizzare gli stessi documenti rispetto a un riferimento fissato. Per semplicità, assumiamo di aver già scelto il tono di riferimento, determinato dal documento di Figura 9.

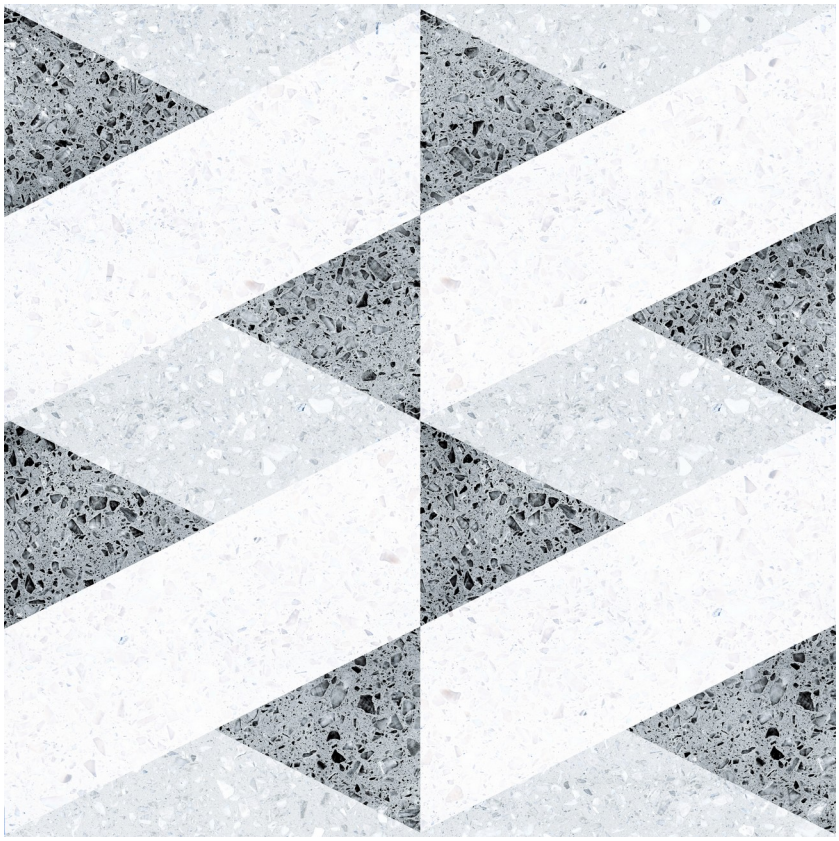

Figura 9: Documento di riferimento, al cui tono devono essere riportati i toni di tutti gli altri.

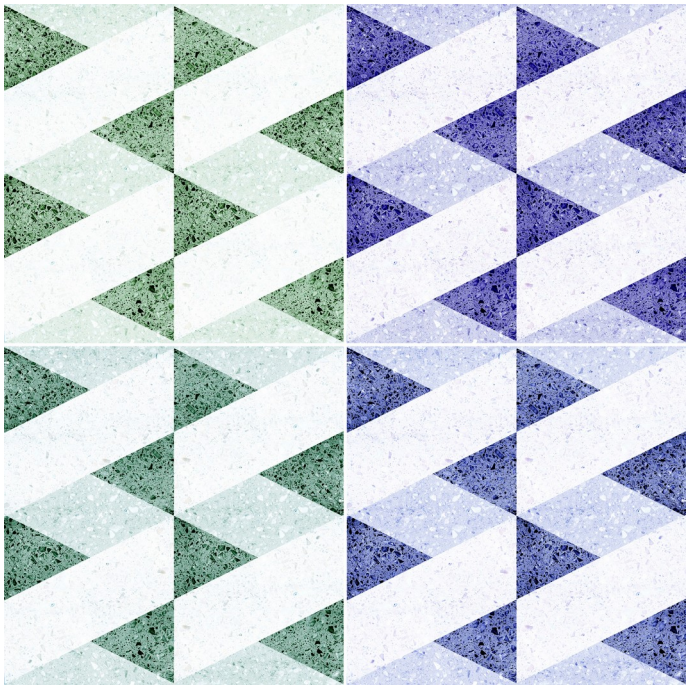

Figura 10: I documenti da riportare a riferimento, denominati 01_TEST_64X64.jpg, 02_TEST_64X64.jpg, 03_TEST_64X64.jpg e 04_TEST_64X64.jpg (risp.)

Si può invocare PDP per effettuare queste operazioni tramite uno script di avvio, richiamato dall'apposito menu di Photoshop, di cui si riporta una possibile implementazione:

```
#include "source/oggetti-minimi/Pipeline.jsx"
#include "source/sovrimpressione-info/FiltroSovrimpressione.jsx"
#include "source/tonalizzazione/FiltroTonalizzazioneCMYK.jsx"

// Salvataggio preferenze Photoshop
var preferenzeRighelli = app.preferences.rulerUnits;

// Definizione e configurazione FiltroSovrimpressione
var configurazioneSovrimpressione = {
    azioneConfigurazione: "FILES NUMERATI",
    setAzioneConfigurazione: "NUMERAZIONE"
};

var margine = 2;
var estrattoreInfo = new EstrattoreCodiceNumericoStandard();
var posizionatoreLivello = new PosizionatoreLivello(margine);
var filtroSovrimpressione = new FiltroSovrimpressione(configurazioneSovrimpressione, estrattoreInfo, posizionatoreLivello);

// Definizione FiltroLetturaTonoCMYK
var tabellaToni = new TabellaToniCMYK(estrattoreInfo);
var scrittoreFile = new ScrittoreTabellaToni();
var filtroLetturaTono = new FiltroLetturaTonoCMYK(tabellaToni, scrittoreFile);

// Definizione FiltroTonalizzazioneCMYK e composizione con FiltroLetturaTonoCMYK
var filtroTonalizzazione = new FiltroTonalizzazioneCMYK(filtroLetturaTono);

// Definizione Pipeline e aggiunta filtri
var pipeline = new Pipeline(app.documents);
pipeline.aggiungiFiltro(filtroSovrimpressione);
pipeline.aggiungiFiltro(filtroTonalizzazione);

// Settaggio preferenze Photoshop alle preferenze di riferimento per PDP, esecuzione pipeline
app.preferences.rulerUnits = Units.CM;
pipeline.esegui();

// Ripristino preferenze Photoshop
app.preferences.rulerUnits = preferenzeRighelli;
```

Figura 11: Script di avvio di PDP.

Una volta aperti i documenti di Figura 10 in Photoshop ed invocato lo script, PDP provvederà a sovrimporre su ciascuno di essi il proprio codice numerico, dopodiché compilerà la tabella dei toni di questi documenti e da questa calcolerà il tono medio. La tabella dei toni e il tono medio sono poi sia visualizzati all'utente tramite una finestra di dialogo sia salvati su un file di testo. A questo punto, tipicamente si utilizzano queste informazioni per scegliere un valore o range di riferimento per ciascun canale, eventualmente consultando il cliente del lotto di piastrelle. In questo caso, avendo ipotizzato di aver già stabilito il tono di riferimento, sarà sufficiente inserire le quantità percentuali di riferimento per ciascun canale tramite le opportune finestre di dialogo, e PDP provvederà autonomamente alla tonalizzazione, il cui risultato è riportato in Figura 13.

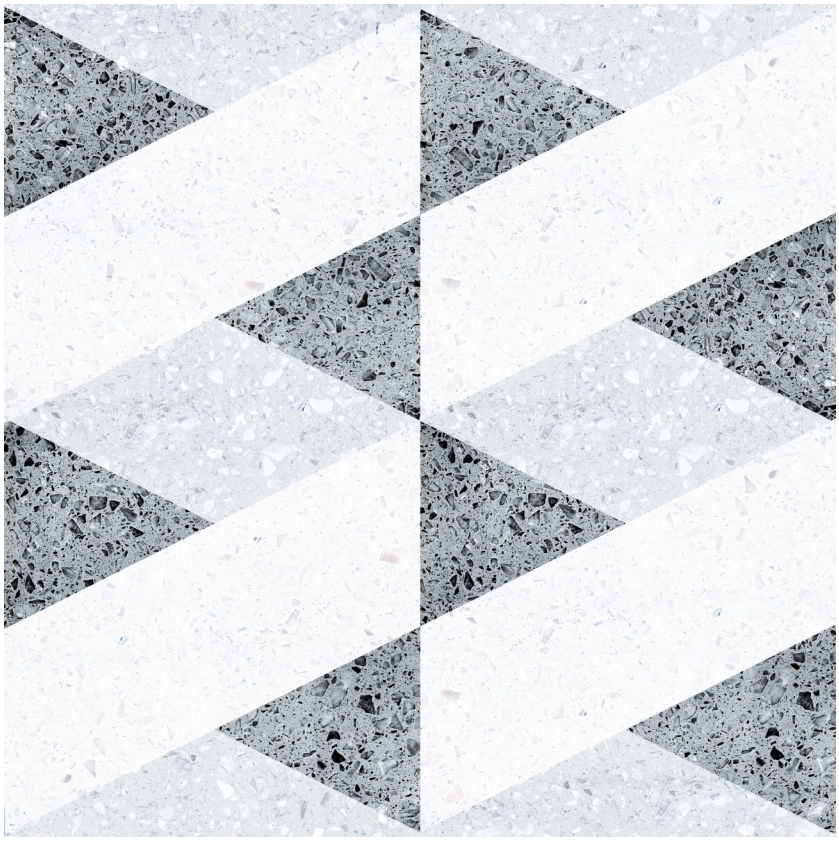

Figura 12: Documento di riferimento, qui riportato per facilitare il confronto con i documenti processati.

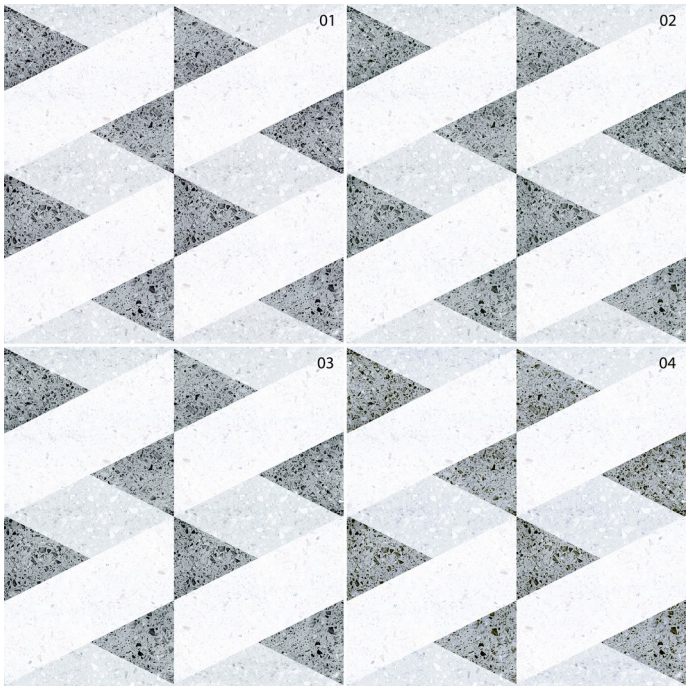

Figura 13: Documenti processati. Notare il codice numerico nell'angolo in alto a destra e, soprattutto, notare come il tono sia stato uniformato a quello di riferimento.

5 Conclusioni

Si noterà forse l'assenza di un capitolo dedicato al testing: questo non deve far pensare che testare il codice non sia stato ritenuto importante, tutt'altro, ma il tempo a disposizione non si è rivelato sufficiente. E' sottinteso che PDP sarà testato in maniera approfondita nelle settimane a venire, prima di essere poi effettivamente utilizzato dall'azienda ceramica di riferimento.

Ciò detto, si chiude il presente elaborato con una breve riflessione sulle conoscenze acquisite nel corso della progettazione e implementazione di PDP. Come si è avuto modo di vedere, sviluppare questo progetto ha richiesto le seguenti conoscenze:

- Conoscenza di JavaScript/ExtendScript.
- Conoscenza di base di Adobe Photoshop e di scripting per Photoshop.
- Conoscenza delle procedure impiegate in un'azienda ceramica nella pianificazione e nel controllo della produzione di piastrelle, relativamente alle operazioni comunemente eseguite su Photoshop.

Prima ancora di pensare alla progettazione di PDP, il sottoscritto ha dovuto recuperare queste conoscenze: questo è stato fatto in parte studiando manuali sugli argomenti (in particolare per quanto riguarda il linguaggio JavaScript, intimamente legato a ExtendScript) e in parte grazie al continuo confronto con un esperto del campo, che si trova quotidianamente ad affrontare i problemi e le limitazioni che PDP si pone di risolvere.

6 Appendici

In questo capitolo si troveranno i principali manuali utilizzati, come riferimento, nella progettazione e implementazione di PDP, con i relativi link per il download. Questi non sono da considerarsi necessari alla comprensione del suo funzionamento, ma piuttosto come un supplemento volto a chiarire in che modo si integri con le astrazioni e funzionalità già offerte da Photoshop.

- Adobe Photoshop CS5 Scripting Guide
  https://www.adobe.com/content/dam/acom/en/devnet/photoshop/scripting/photoshop_cs5_scripting_guide.pdf
- Adobe Photoshop CS5 JavaScript Scripting Reference
  https://www.adobe.com/content/dam/acom/en/devnet/photoshop/scripting/photoshop_cs5_javascript_ref.pdf
- JavaScript Tools Guide CC
  https://extendscript.docsforadobe.dev/_/downloads/en/latest/pdf/