

Università degli Studi di Trento
Dipartimento di Economia e Management

elm-foldbook:

un prototipo di foglio di calcolo multidimensionale
integrato con documenti matematici
per la didattica e la ricerca in finanza aziendale

Nota tecnica sul prodotto della ricerca di tipo *Software*

Luca Erzegovesi

Versione del software	0.7 (16 dicembre 2024); rilasci pubblici 0.7.1–0.7.2 (2026)
Periodo di sviluppo	giugno – dicembre 2024
Repository	https://github.com/lerzegov/elm-foldbook
Motore di calcolo	https://github.com/lerzegov/elm-interpreter-fork
DOI	10.5281/zenodo.23056100
DOI del motore	10.5281/zenodo.23056240
SWHID (tag v0.7)	swh:1:rel:1a45a295993b1acee16b332b8dfa3aea2ad3d300
Licenza	BSD 3-Clause

Trento, 14 ottobre 2026

Scheda del prodotto

Titolo	elm-foldbook: un prototipo di foglio di calcolo multidimensionale integrato con documenti matematici per la didattica e la ricerca in finanza aziendale
Tipologia	Software
Autore	Luca Erzegovesi, Università di Trento
Sviluppo	giugno – dicembre 2024 (tag v0.0–v0.7)
Versione di riferimento	0.7, tag v0.7, commit 05d3e76, 16/12/2024
Rilascio pubblico	0.7.1–0.7.2 (2026): impacchettamento e documentazione; codice applicativo identico alla 0.7
Linguaggi	Elm 0.19.1; TypeScript/JavaScript per i componenti web
Dimensioni	circa 19.000 righe di codice Elm e 1.500 di TypeScript/JavaScript scritte dall'autore, più l'interprete di terzi su cui il motore si fonda
Licenza	BSD 3-Clause; componenti di terzi con licenze MIT, BSD-3, Apache-2.0
Stato	prototipo di ricerca, eseguibile; proseguito nel progetto Impromptu
Disponibilità	repository GitHub pubblico; archivio Software Heritage; deposito Zenodo con DOI
Identificativi	DOI 10.5281/zenodo.23056100 (rilascio 0.7.2); SWHID del tag v0.7: swh:1:rel:1a45a295993b1acee16b332b8dfa3aea2ad3d300
Motore di calcolo	fork pubblico su GitHub dal 29/07/2024, ultimo aggiornamento 18/12/2024; DOI 10.5281/zenodo.23056240

Sommario

elm-foldbook è un'applicazione web, scritta nel linguaggio funzionale Elm, che sperimenta un modo diverso di costruire modelli finanziari. I dati sono organizzati in *dataset* con dimensioni dotate di nome (azienda, anno, voce di bilancio, scenario). Le formule si scrivono una volta per un'intera porzione del dataset, non cella per cella. I modelli vivono dentro documenti che combinano testo, equazioni in notazione matematica e fogli di calcolo interattivi. Il motore di calcolo estende un interprete del linguaggio Elm scritto in Elm (elm-interpreter di Leonardo Taglialegne) con valori multidimensionali e aritmetica vettoriale. Il software è stato sviluppato tra giugno e dicembre 2024 come progetto pilota. Le sue soluzioni e i suoi limiti hanno guidato la progettazione di Impromptu, l'ambiente di modellazione multidimensionale che l'autore sviluppa dal 2026, con un front end Elm derivato da elm-foldbook e un motore di calcolo in Julia.

Abstract

elm-foldbook is a browser application written in Elm that explores an alternative way of building financial models. Data are organised in datasets with named dimensions. Formulas are written once for whole array slices. Models live inside documents that mix prose, mathematical notation and interactive spreadsheets. The calculation engine extends an Elm interpreter written in Elm (L. Taglialegne's elm-interpreter) with multidimensional values and array arithmetic. Developed between June and December 2024 as a pilot, it shaped the design of Impromptu, the author's current multidimensional modelling environment, whose Elm front end derives from elm-foldbook and whose engine runs in Julia.

Indice

Scheda del prodotto	1
1 Il problema affrontato	3
1.1 I limiti del foglio di calcolo a celle	3
1.2 Il bisogno didattico	3
1.3 Il bisogno di ricerca	3
1.4 Obiettivi del prototipo	3
2 Stato dell'arte	4
2.1 Fogli di calcolo multidimensionali	4
2.2 OLAP e tabelle pivot	4
2.3 Array etichettati nel calcolo scientifico	4
2.4 Evoluzione dei fogli di calcolo tradizionali	4
2.5 Documenti computazionali	4
2.6 Elm ed elm-interpreter	4
2.7 Lo spazio vuoto	5
3 Contributo originale	5
3.1 Cosa è originale e cosa è ereditato	5
3.2 I tre contributi principali	5
3.3 Cronologia dello sviluppo	6
4 Architettura	6
4.1 Visione d'insieme	6
4.2 Il modello dati multidimensionale	7
4.3 Il linguaggio delle formule	7
4.4 Il motore di calcolo	7
4.5 Gli editor	8
4.6 Integrazione fra testo, notazione matematica e fogli	8
4.7 Scelte tecnologiche	8
5 Esempi d'uso	9
5.1 Conto economico multi-azienda e riorganizzazione pivot	9
5.2 Analisi <i>what-if</i> con ricalcolo a catena	10
5.3 Scrivere formule con l'aiuto dell'editor	10
5.4 Dalla formula del manuale alla funzione riusabile: le obbligazioni	11
5.5 Valutazione dell'equity: FCFE ed extra-rendimenti	11
5.6 Il documento come sorgente	12
5.7 Usi previsti	13
6 Limiti e sviluppi: da elm-foldbook a Impromptu	13

6.1	Limiti del prototipo	13
6.2	Lezioni apprese e decisione architettuale	14
6.3	Impromptu	14
6.4	Stato attuale e diffusione	14
7	Disponibilità, licenza e riproducibilità	15
7.1	Dove si trova	15
7.2	Licenza	15
7.3	Eseguire e ricompilare	15
7.4	Nota sulla cronologia e sulla versione 0.7.1	15
	Ringraziamenti	16
	Riferimenti bibliografici	16
A	Moduli del software	17
B	Esempi di sorgente	18

1 Il problema affrontato

1.1 I limiti del foglio di calcolo a celle

Il foglio di calcolo elettronico è lo strumento dominante della modellistica finanziaria: piani economico-finanziari, valutazioni d'azienda, analisi di scenario. La sua forza è l'immediatezza. Il modello e i suoi numeri coincidono con una griglia che l'utente vede e modifica direttamente. La stessa caratteristica ne determina i limiti quando il modello cresce:

- **Struttura implicita.** Le dimensioni del problema (aziende, anni, voci, scenari) non esistono come oggetti: sono convenzioni di layout, righe e colonne etichettate a mano. Aggiungere un anno o uno scenario significa copiare blocchi di celle e correggere riferimenti.
- **Formule ripetute.** La stessa relazione economica, ad esempio $EBITDA = \text{margine di contribuzione} - \text{spese generali}$, è replicata in decine o centinaia di celle. Una modifica va propagata a mano e un'eccezione locale passa inosservata. La letteratura sugli errori nei fogli di calcolo documenta da tempo la frequenza di questi problemi (Panko 1998; Hermans et al. 2015).
- **Presentazione e calcolo legati.** Per vedere i dati da un'altra prospettiva, ad esempio anni sulle righe e voci sulle colonne, bisogna costruire un secondo prospetto o una tabella pivot separata dal modello.

1.2 Il bisogno didattico

Nei corsi di finanza aziendale la stessa idea vive in due forme che raramente si incontrano: la *formula* del manuale (il valore attuale dei flussi di cassa, il modello a extra-rendimenti) e il *modello numerico* che lo studente costruisce nel foglio di calcolo. Il passaggio dall'una all'altra è il punto in cui si concentrano gli errori e in cui si perde il senso economico. Uno strumento che mostri nello stesso documento la notazione matematica, il testo che la spiega e il modello che la calcola rende visibile la relazione fra teoria e implementazione.

1.3 Il bisogno di ricerca

Nella ricerca applicata, per esempio sulla pianificazione finanziaria o sulla valutazione di portafogli obbligazionari, i modelli hanno naturalmente più dimensioni. Serve poterli esplorare riorientando le dimensioni, documentarli in modo riproducibile e modificarne la struttura senza riscriverli.

1.4 Obiettivi del prototipo

elm-foldbook è nato per verificare, con un prototipo funzionante, che fosse possibile:

1. rappresentare un modello come insieme di dataset con dimensioni condivise e dotate di nome;
2. scrivere formule a livello di array, riusando un vero linguaggio di programmazione invece di inventare una sintassi ad hoc;
3. visualizzare e riorganizzare i dataset in una griglia di tipo pivot, con trascinamento delle dimensioni;
4. incorporare fogli vivi ed equazioni in documenti di testo;
5. fare tutto questo in un'applicazione web leggera, senza server di calcolo.

2 Stato dell'arte

2.1 Fogli di calcolo multidimensionali

L'idea di separare dati, formule e presentazione risale a *Lotus Improv* (1991), che organizzava i dati in categorie con nome e scriveva le formule in termini di categorie ($Ricavi = Prezzo * Quantità$) invece che di indirizzi di cella. Le viste si riorganizzavano trascinando le categorie. Improv non ebbe successo commerciale, ma le sue idee sono state riprese da *Quantrix Modeler*, tuttora usato nella pianificazione finanziaria (Quantrix, s.d.). Nella stessa linea si collocano le piattaforme di *connected planning* diffuse nelle imprese (Anaplan, Pigment e simili), che uniscono modello multidimensionale, collaborazione e integrazione con i sistemi aziendali. Sono prodotti proprietari, pensati per l'uso aziendale e non per l'insegnamento, e il loro linguaggio di formula non è estensibile dall'utente.

2.2 OLAP e tabelle pivot

Le basi dati multidimensionali (OLAP) (Codd et al. 1993) e le tabelle pivot dei fogli di calcolo offrono la navigazione per dimensioni e l'aggregazione, ma sono strumenti di *analisi*. Le formule che definiscono un modello di pianificazione restano fuori dal cubo o sono limitate a misure calcolate.

2.3 Array etichettati nel calcolo scientifico

Nel calcolo scientifico gli array con dimensioni e coordinate etichettate sono uno standard: *pandas* (McKinney 2010), *xarray* (Hoyer e Hamman 2017), e in Julia *DimensionalData.jl*. Selezione per coordinata (`loc`), per posizione (`iloc`) e allineamento automatico nelle operazioni sono proprio le operazioni che un modello finanziario multidimensionale richiede. Questi strumenti però sono librerie per programmatori, senza interfaccia a griglia.

2.4 Evoluzione dei fogli di calcolo tradizionali

Microsoft Excel ha introdotto gli *array dinamici* e la funzione LAMBDA, che avvicinano il foglio di calcolo a un linguaggio funzionale (Peyton Jones et al. 2003; Gordon et al. 2021). Il modello resta però legato agli indirizzi di cella: le dimensioni non diventano oggetti del modello.

2.5 Documenti computazionali

La *literate programming* di Knuth (Knuth 1984) e i notebook (Jupyter (Kluyver et al. 2016), Quarto, Observable) uniscono testo, codice e risultati. Gli editor matematici per il web, in particolare MathLive con il Cortex Compute Engine (Gourdol, s.d.), permettono di scrivere e valutare espressioni in notazione matematica. Nei notebook però il modello è codice, non una griglia che l'utente di un foglio di calcolo riconosce e modifica.

2.6 Elm ed elm-interpreter

Elm (Czaplicki e Chong 2013) è un linguaggio funzionale puro e tipizzato per applicazioni web. Con la sua architettura a messaggi (*The Elm Architecture*) si scrivono interfacce complesse e affidabili. elm-interpreter di Leonardo Taglialegne (Taglialegne 2023) è un interprete di Elm scritto in Elm: analizza il sorgente con la libreria `elm-syntax` e lo valuta nel browser, con la libreria standard `elm/core` rigenerata per l'interprete.

2.7 Lo spazio vuoto

Nessuno degli strumenti citati combina, in un'applicazione web leggera e aperta: (a) un modello multidimensionale con dimensioni dotate di nome e una griglia pivot modificabile; (b) formule a livello di array scritte in un linguaggio di programmazione completo; (c) documenti che contengono insieme testo, notazione matematica valutabile e fogli vivi. `elm-foldbook` è stato costruito per esplorare questa combinazione.

3 Contributo originale

3.1 Cosa è originale e cosa è ereditato

Il software è composto da due repository. L'applicazione (`elm-foldbook`) contiene interfaccia, documenti ed editor. Il motore di calcolo è un fork di `elm-interpreter` (`elm-interpreter-fork`, pubblico dal 29 luglio 2024), incluso nel rilascio 0.7.1 nella cartella `vendor/`. La Tabella 1 distingue il contributo dell'autore dal codice ereditato, contando le righe con `git blame` sulle revisioni di riferimento.

Tabella 1: Righe di codice sorgente per origine (revisioni di dicembre 2024).

	L. Erzegovesi	Ereditate
Applicazione <code>elm-foldbook</code> (Elm)	11.848	–
Applicazione <code>elm-foldbook</code> (TypeScript/JavaScript)	≈ 1.530	–
Test dell'interprete originale (non usati dall'app)	–	2.122
Motore: fork di <code>elm-interpreter</code> (<code>src/</code>)	7.102	4.788
Motore: <code>elm/core</code> rigenerata (<code>generated/</code>)	–	codice generato
Totale	≈ 20.500	≈ 6.900

Il conteggio delle righe non misura il valore intellettuale. L'interprete di Taglialegne è la base senza la quale il motore non esisterebbe. Il contributo originale sta in ciò che vi è stato costruito sopra e intorno. I moduli sono elencati nell'appendice A.

3.2 I tre contributi principali

1. **Un modello multidimensionale dentro un interprete.** Il modello dati `XModel` (dimensioni condivise, dataset, array piatti con strides, selezione `loc/iloc`, editing strutturale) è stato innestato nell'interprete Elm come nuovo tipo di valore (`DataAr`). Le operazioni aritmetiche dell'interprete sono state estese al *broadcasting* e all'allineamento per coordinate. Un nome di intervallo come `ce_valore_ebitda` diventa così una variabile Elm valida, e l'interprete ne carica il contenuto solo quando serve. Il risultato è un linguaggio di formula tipizzato e completo (funzioni, `let`, `case`, funzioni di ordine superiore) senza scrivere un parser nuovo.
2. **Una griglia pivot modificabile.** Intestazioni gerarchiche di righe e colonne, vassoi pagina/righe/colonne con trascinamento delle dimensioni, modifica di celle, intestazioni e struttura, ricalcolo immediato.
3. **Documenti con modelli vivi.** Documenti in `elm-markup` che contengono blocchi di testo, blocchi Sheet (un dataset interattivo completo) e blocchi Equation (LaTeX modificabile con MathLive e valutato dal Compute Engine), con valori numerici modificabili nel testo.

3.3 Cronologia dello sviluppo

La Tabella 2 riporta le tappe, documentate dai tag del repository. La data di inizio è attestata dai metadati del server GitHub (creazione del repository il 19/06/2024) e da tre commit dello stesso giorno firmati e verificati da GitHub. Quella di fine è attestata dall'ultimo push sul server (16/12/2024), registrato il 29/09/2026 prima della pubblicazione. Il fork con il motore di calcolo è pubblico dal 29/07/2024. Il fork di elm-markup è stato pubblicato sul registro ufficiale dei pacchetti Elm il 17/07/2024.

Tabella 2: Versioni di elm-foldbook.

Tag	Data	Contenuto
v0.0–v0.1	19/06/2024	Progetto derivato da elm-interpreter; home page foldBook
v0.2	26/06/2024	Nomi di intervallo, parsing delle formule, suggerimenti nell'editor
v0.3	01/07/2024	CodeMirror 6; ricalcolo fra dataset
0.4–0.5	29/07/2024	Motore e XModel spostati nel fork pubblico dell'interprete
0.6	12/08/2024	Documenti elm-markup con blocchi foglio; editor MathLive
v0.7	16/12/2024	Revisione della gestione della struttura della griglia (versione di riferimento)
v0.7.1	29/09/2026	Rilascio pubblico: motore incluso, licenze, documentazione
v0.7.2	30/09/2026	README aggiornato, metadati Zenodo; deposito con DOI

4 Architettura

4.1 Visione d'insieme

elm-foldbook è un'applicazione a pagina singola (Browser.application) che gira interamente nel browser. La Figura 1 mostra i componenti. Il programma Elm comunica con due componenti web in JavaScript attraverso le *porte* Elm: l'editor di codice CodeMirror 6 e l'editor matematico MathLive.

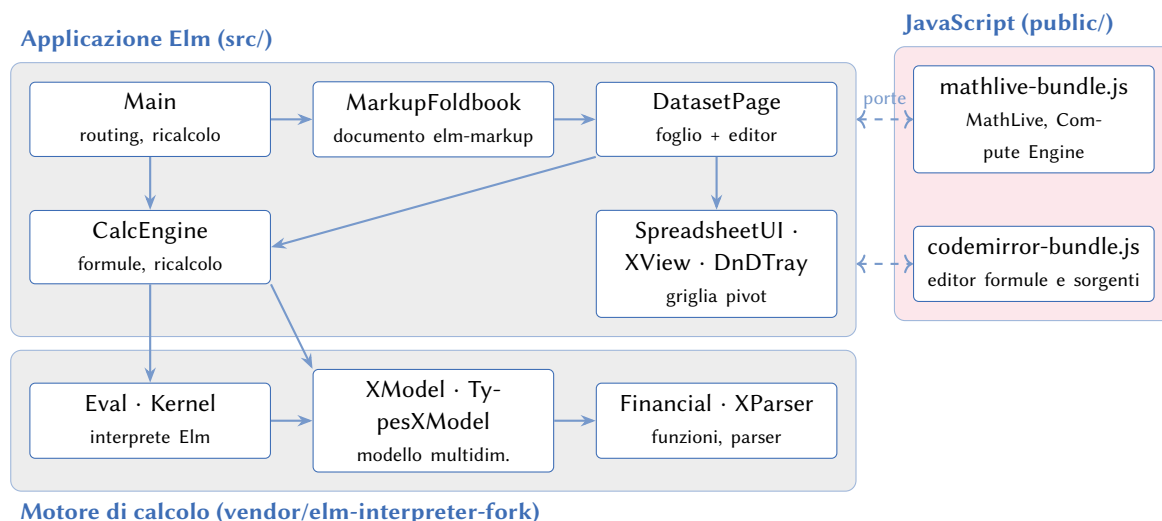


Figura 1: Componenti di elm-foldbook 0.7. Le frecce continue indicano chiamate fra moduli Elm, quelle tratteggiate le porte verso i componenti JavaScript (MarkupFoldbook usa MathLive per le equazioni; DatasetPage e SpreadsheetUI usano CodeMirror).

4.2 Il modello dati multidimensionale

Il modello (XModel) contiene un insieme di *dimensioni condivise*, ciascuna con un elenco ordinato di coordinate (per esempio anno: a1, a2, a3), e un insieme di *dataset*. Ogni dataset:

- sceglie un sottoinsieme ordinato delle dimensioni del modello; il conto economico Ce, ad esempio, usa azienda × voce × anno;
- contiene una o più variabili (DataArray), memorizzate come array piatti di numeri e, in parallelo, di testi, in ordine *row-major* con passi (*strides*) calcolati dalle dimensioni;
- contiene il sorgente delle proprie formule, sotto forma di modulo Elm.

Le variabili di un dataset formano una pseudo-dimensione aggiuntiva, dato, che si sposta nei vassoi come le altre. Nella griglia si possono quindi disporre le variabili sulle righe, sulle colonne o nella pagina.

La selezione avviene per coordinata o per posizione, singola, per intervallo o per lista, come in pandas e xarray (loc, iloc). Le funzioni di editing strutturale aggiungono, inseriscono, rinominano ed eliminano coordinate, dimensioni e variabili, e riallineano i dati esistenti alla nuova forma.

4.3 Il linguaggio delle formule

Le formule di un dataset sono un modulo Elm. Ogni dichiarazione di primo livello ha come nome l'intervallo di destinazione, secondo la grammatica

dataset__variabile_coordinata_coordinata...

Le coordinate vengono attribuite automaticamente alla loro dimensione tramite un dizionario inverso. Se una coordinata compare in più dimensioni si qualifica come dimDOTcoord. Il modulo del conto economico, per esempio, contiene:

```
module Ce exposing (..)
taxRate = 0.4
costoVen = ce__valore_ricavi * 0.48 * macro__cambioUsdEur_base
ce__valore_margContrib = mySub ce__valore_ricavi ce__valore_costoVen
ce__valore_ebitda = ce__valore_margContrib - ce__valore_speseVGA
ce__valore_ebit = ce__valore_ebitda - ce__valore_amm
ce__valore_tax = ce__valore_ebit * taxRate
ce__valore_unlevNetIncome = ce__valore_ebit - ce__valore_tax
```

Ogni riga calcola una voce per *tutte* le aziende e *tutti* gli anni. La variabile macro__cambioUsdEur_base proviene da un altro dataset (Macro, con dimensioni scenario × anno). Viene selezionata allo scenario base e allineata automaticamente sulla dimensione anno del conto economico. mySub è una normale funzione Elm definita nello stesso modulo: il linguaggio di formula è l'intero linguaggio Elm.

4.4 Il motore di calcolo

Il motore è l'interprete di Taglialegne con le estensioni dell'autore:

- **Risoluzione pigra dei nomi.** Quando l'interprete incontra un nome che non è né una funzione né un valore definito, lo interpreta come nome di intervallo e inserisce nell'ambiente di valutazione l'array corrispondente, o la sua sezione.

- **Aritmetica vettoriale.** Le funzioni matematiche di base e gli operatori aritmetici del kernel accettano array: scalare con array, array con scalare, array con array. Se due array hanno dimensioni diverse, i dati vengono rimappati per coordinate e le combinazioni mancanti diventano valori non numerici (NaN).
- **Un ambiente per dataset.** Ogni dataset ha un proprio ambiente dell'interprete, costruito dal modulo delle formule. Una formula si valuta senza rileggere il modulo.
- **Scrittura dei risultati.** Uno scalare viene esteso a tutto l'intervallo di destinazione. Una lista o un array vengono controllati in lunghezza. Un array multidimensionale viene scritto nella variabile di destinazione.
- **Funzioni finanziarie:** prezzo e rendimento di obbligazioni (con il metodo di Newton-Raphson), VAN, TIR, Black-Scholes, Garman-Kohlhagen.

Il **ricalcolo** valuta le formule nell'ordine in cui sono dichiarate. Un dataset può dichiarare i dataset che dipendono da lui (`dependentDatasetRefs = [ "Ce" ]`), che vengono accodati e ricalcolati dopo di lui. La modifica di una cella accoda il dataset corrente. È una soluzione semplice e deterministica. Non costruisce però un grafo delle dipendenze e non rileva i riferimenti circolari (Sezione 6).

4.5 Gli editor

Griglia pivot. La griglia è costruita con CSS grid e intestazioni fisse. Le intestazioni di righe e colonne sono alberi (HeaderTree) calcolati dalle dimensioni disposte nei vassoi, con celle che si estendono su più righe o colonne. Tre vassoi (pagina in grigio, righe in azzurro, colonne in rosa) contengono i *token* delle dimensioni, che si trascinano da un vassoio all'altro (Figura 3). Le dimensioni nel vassoio pagina mostrano un menu per scegliere la coordinata. Con un doppio clic si modificano celle e intestazioni. I comandi strutturali aggiungono e tolgono coordinate e dimensioni.

Editor delle formule. Il modulo delle formule si modifica in CodeMirror 6. Mentre si scrive un nome di intervallo, l'editor chiede all'applicazione le coordinate compatibili con il prefisso già scritto, escludendo le dimensioni già usate, e le propone come completamento (Figura 5).

Editor matematico. Le equazioni sono componenti web basati su MathLive. Il Cortex Compute Engine le valuta. L'autore lo ha esteso con macro finanziarie ($\backslash ytm$, $\backslash FV$, $\backslash CPN$, $\backslash PV$) e con la possibilità di definire funzioni ($\backslash coloneqq$) da riusare in altre equazioni. La visualizzazione include la lettura vocale in italiano.

4.6 Integrazione fra testo, notazione matematica e fogli

I documenti sono scritti in *elm-markup*, un linguaggio di markup a blocchi tipizzati (Griffith 2018). L'autore ne ha pubblicato un fork sul registro dei pacchetti Elm. I blocchi disponibili sono titoli, testo, liste, immagini, codice, Equation e Sheet; all'interno del testo si possono inserire equazioni, fogli e valori numerici modificabili. Un blocco Sheet incorpora un dataset completo, con griglia, vassoi ed editor delle formule. Il documento si modifica a blocchi o come sorgente (Figura 8), e si salva come file locale. Un indice laterale permette di navigare fra le sezioni.

4.7 Scelte tecnologiche

La scelta di Elm offre tre vantaggi: un'architettura dell'interfaccia prevedibile, un compilatore che esclude intere classi di errori a tempo di esecuzione e, soprattutto, un interprete dello stesso linguaggio già disponibile. Proprio questo ha permesso di usare Elm anche come linguaggio delle formule. L'applicazione si compila con il compilatore Elm (o `elm-watch` durante lo sviluppo). Il componente MathLive è impacchettato con Rollup. La libreria `elm/core` per l'interprete è rigenerata con `elm-codegen`.

5 Esempi d'uso

Il documento di dimostrazione caricato all'avvio contiene un piccolo modello di pianificazione (finPlan) con tre dataset: anagrafica delle aziende (Az), conto economico di due aziende in tre anni (Ce) e scenari macroeconomici di inflazione e cambio (Macro). Seguono equazioni valutabili sulla valutazione di obbligazioni. Un secondo documento, Equations.emu, tratta i modelli di valutazione dell'equity.

 **Welcome to elm-foldBook**

[View Source](#) [Save source from markup](#) [Show Popups](#) [All block types -> only sheets](#)

Fogli di calcolo di prova

- Aziende
- Conto economico
- Scenario

Equazioni in formato LaTeX con MathLive

Custom functions and variables

Calcolo del prezzo di un'obbligazione con funzioni

Equations with matrices

Fogli di calcolo di prova

Provvisoriamente alla fine di XModel.elm

Aziende

azienda

dato anno

azienda	nome			settore			nrDip			dato anno
	a1	a2	a3	a1	a2	a3	a1	a2	a3	
alfa	Alfa SpA	Alfa SpA	Alfa SpA	Industria	Industria	Industria	10.0	12.0	20.0	
beta	Beta SpA	Beta SpA	Beta SpA	Commercio	Commercio	Commercio	20.0	24.0	40.0	

Conto economico

azienda voce

dato anno

azienda	voce	valore			dato anno
		a1	a2	a3	
alfa	ricavi	100.0	209.0	300.0	
	costoVen	52.8	100.3	216.0	
	margContrib	47.2	108.7	84.0	
	speseVGA	6.0	17.0	18.0	
	ebitda	41.2	91.7	66.0	
	amm	12.0	29.0	16.0	
	ebit	29.2	62.7	50.0	
	tax	11.7	25.1	20.0	
	unlevNetIncome	17.5	37.6	30.0	
	beta	ricavi	1,007.0	2,007.0	3,007.0

Figura 2: Il documento di dimostrazione: indice laterale, testo e due fogli vivi (anagrafica delle aziende e conto economico) con i rispettivi vassoi delle dimensioni.

5.1 Conto economico multi-azienda e riorganizzazione pivot

Il dataset Ce ha tre dimensioni (azienda, voce, anno) e una variabile (valore). Nella vista iniziale (Figura 3, in alto) azienda e voce stanno sulle righe e anno sulle colonne. Trascinando azienda nel vassoio pagina, anno sulle righe e voce sulle colonne si ottiene, senza toccare il modello, il conto economico dell'azienda *alfa* con gli anni sulle righe (Figura 3, in basso). Il menu nel vassoio pagina seleziona l'azienda.

Conto economico

azienda

voce

dato

anno

azienda	voce	valore			dato
		a1	a2	a3	anno
alfa	ricavi	100.0	209.0	300.0	
	costoVen	52.8	100.3	216.0	
	margContrib	47.2	108.7	84.0	
	speseVGA	6.0	17.0	18.0	
	ebitda	41.2	91.7	66.0	
	amm	12.0	29.0	16.0	
	ebit	29.2	62.7	50.0	
	tax	11.7	25.1	20.0	
	unlevNetIncome	17.5	37.6	30.0	
	beta	ricavi	1,007.0	2,007.0	3,007.0
costoVen		531.7	963.4	2,165.0	
margContrib		475.3	1,043.6	842.0	
speseVGA		67.0	127.0	187.0	
ebitda		408.3	916.6	655.0	
amm		127.0	247.0	167.0	
ebit		281.3	669.6	488.0	
tax		112.5	267.9	195.2	
unlevNetIncome		168.8	401.8	292.8	

Conto economico

azienda

alfa

voce

dato

anno

anno	ricavi	costoVen	margContrib	speseVGA	ebitda	amm	ebit	tax	unlevNetIncome	voce
	valore	valore	valore	valore	valore	valore	valore	valore	valore	dato
a1	100.0	52.8	47.2	6.0	41.2	12.0	29.2	11.7	17.5	
a2	209.0	100.3	108.7	17.0	91.7	29.0	62.7	25.1	37.6	
a3	300.0	216.0	84.0	18.0	66.0	16.0	50.0	20.0	30.0	

Figura 3: Lo stesso dataset in due viste. In alto: azienda e voce sulle righe, anni sulle colonne. In basso, dopo tre trascinatori: azienda nel vassoio pagina (menu *alfa*), anni sulle righe, voci sulle colonne.

5.2 Analisi *what-if* con ricalcolo a catena

Le voci derivate (costo del venduto, margine, EBITDA, EBIT, imposte, reddito netto *unlevered*) sono calcolate dalle formule del modulo Ce. Il costo del venduto dipende anche dal cambio USD/EUR del dataset Macro. Se si modificano i ricavi dell'azienda alfa nel primo anno da 100 a 150, l'intera catena viene ricalcolata (Figura 4): il costo del venduto passa da 52,8 a 79,2, l'EBITDA da 41,2 a 64,8, il reddito netto da 17,5 a 31,7.

Conto economico

azienda voce

dato anno

azienda	voce	valore			dato
		a1	a2	a3	anno
alfa	ricavi	100.0	209.0	300.0	
	costoVen	52.8	100.3	216.0	
	margContrib	47.2	108.7	84.0	
	speseVGA	6.0	17.0	18.0	
	ebitda	41.2	91.7	66.0	
	amm	12.0	29.0	16.0	
	ebit	29.2	62.7	50.0	
	tax	11.7	25.1	20.0	
	unlevNetIncome	17.5	37.6	30.0	

Conto economico

azienda voce

dato anno

azienda	voce	valore			dato
		a1	a2	a3	anno
alfa	ricavi	150.0	209.0	300.0	
	costoVen	79.2	100.3	216.0	
	margContrib	70.8	108.7	84.0	
	speseVGA	6.0	17.0	18.0	
	ebitda	64.8	91.7	66.0	
	amm	12.0	29.0	16.0	
	ebit	52.8	62.7	50.0	
	tax	21.1	25.1	20.0	
	unlevNetIncome	31.7	37.6	30.0	

(a) Modello originale

(b) Ricavi di alfa nel primo anno portati a 150

Figura 4: Analisi *what-if*: le celle derivate vengono ricalcolate dalle formule a livello di array.

5.3 Scrivere formule con l'aiuto dell'editor

L'editor delle formule mostra il modulo del dataset con evidenziazione della sintassi Elm. Nella Figura 5 l'utente ha iniziato a scrivere `ce__valore_beta_` e l'editor propone tutte le coordinate che possono completare il nome: gli anni e le voci del conto economico dell'azienda beta.

Formula Editor

```

1 module Ce exposing (..)
2 -- getExprDataArray errors on taxRate, browser hangs; cyclic update may stall?!
3 -- error in getExprDataArray are overcome by parsing the module
4 -- check cycle in calcExpressionToXModel
5 mySub : Float -> Float -> Float -- parsing does not catch declaration errors
6 mySub val1 val2 = val1 - val2 -- errors are raised in range parsing
7 taxRate : Float -- declaring a constant expr is not needed, no longer parsing errors
8 taxRate = 0.4
9 -- mapping between datasets thanks to order swap in evalNonVariant before
10 -- executing Kernel.twoNumbers when a binary func is applied
11 costoVen = ce_valore_ricavi * 0.48 * macro_cambioUsdEur_base -- uses expansion of rangeName from getExprDataArray
12 ce_valore_beta_a1 _valore_ricavi ce_valore_costoVen
13 ce_valore_beta_a2 rgContrib - ce_valore_speseVGA
14 ce_valore_beta_a3 da - ce_valore_amm
15 ce_valore_beta_amm * taxRate
16 ce_valore_beta_costoVen alore_ebit - ce_valore_tax
17 ce_valore_beta_ebit
18 ce_valore_beta_ebitda
19 ce_valore_beta_margContrib

```

Figura 5: L'editor delle formule del dataset Ce con il completamento dei nomi di intervallo.

5.4 Dalla formula del manuale alla funzione riusabile: le obbligazioni

Nella sezione sulle equazioni (Figura 6) il prezzo di un titolo con cedola è scritto in notazione matematica e valutato. La somma esplicita dei flussi scontati con cedola 6, rimborso 100, sette anni e tasso del 3% vale 118,69. La stessa formula, assegnata a una funzione Psum con \coloneqq, diventa riusabile: Psum(10, 4, 100, 0.02) restituisce 130,46. Lo studente vede nello stesso documento la formula del manuale, la sua generalizzazione in funzione e il risultato numerico.

Custom functions and variables

put args after the function macro without () and , between args

$$Psum = \sum_{j=1}^n \frac{c}{(1+y)^j} + \frac{v}{(1+y)^n} \text{ empty result}$$

NB eqnBlock ha bisogno di latex code indentato con 4 spazi. Non servono delimitatori di multiline string "" ""

Calcolo prezzo con formula esplicita

$$\sum_{n=1}^7 \frac{6}{(1+0.03)^n} + \frac{100}{(1+0.03)^7}$$

118.69084886566462

Calcolo del valore attuale di un'obbligazione con cedola annuale e rimborso del capitale alla scadenza. Il valore attuale di un' dei valori attuali delle cedole e del rimborso del capitale alla scadenza. La formula è la seguente:

value assignment (non più con multiline expression)

N=5

5

costoVen

Figura 6: Equazioni MathLive valutabili: definizione della funzione Psum (valore attuale di un'obbligazione) e calcolo esplicito del prezzo.

5.5 Valutazione dell'equity: FCFE ed extra-rendimenti

Il documento Equations.emu (Figura 7) è un esempio di materiale didattico. Unisce i fogli del modello a una trattazione in italiano dei modelli di attualizzazione del *free cash flow to equity*, con valore terminale di Gordon, e dei modelli a extra-rendimenti di tipo EVA a due stadi. Il paragrafo iniziale

mostra valori modificabili nel testo: ricavi e costo del venduto si modificano direttamente nella frase, e il margine di contribuzione calcolato si aggiorna.

Prova di *value*

Ho inserito qui valori editabili in grigio che diventa azzurrino in modifica e un valore calcolato in giallo. Per ditta clic su valore, per finire clic fuori (no Enter perché è sempre in input mode, CancelEditValue interrompe soltanto aggiornamento env.values). Possono accettare soltanto numeri: La Alfiero SpA nel 2023 ha realizzato ricavi per [1,000.0] euro, con un costo del venduto di [550.0] euro, producendo un **margine di contribuzione: 450.0**.

Il corsivo non funziona in un paragrafo che comprende elementi inline.

Modelli di attualizzazione del *free cash flow to equity*

Sono un'estensione dei DDM nella quale si modella il *free cash flow to equity* (FCFE), che è il flusso di cassa residuale nei confronti degli azionisti generato dalla gestione. Si ha in tal modo la possibilità di pianificare in maniera più articolata i flussi di cassa a carattere non ricorrente (ad esempio per aumenti di capitale e per rimborso o riacquisto di azioni). Il procedimento di calcolo è molto simile a quello dei DDM: si sostituiscono ai dividendi unitari gli FCFE attesi per calcolare il valore corrente di *equity* E_0

0

in luogo del prezzo dell'azione, poi ottenuto dividendo E_0

empty result

per il numero di azioni. Il tasso di rendimento utilizzato per scontare i flussi è lo stesso. La formula è la seguente:

$$E_0 = \sum_{t=1}^T \frac{FCFE_t}{(1+r_E)^t} + \frac{E_T}{(1+r_E)^T}$$

0

$$E_T = \frac{FCFE_{T+1}}{r_E - g} + \frac{\sum_{j=1}^J \log(j)}{245}$$

0

Modelli di *excess return*

Sono riconducibili all'approccio EVA *Economic Value Added* e spiegano il valore dell'*equity* come somma del suo valore contabile rettificato (BV

0

è di una componente di *equity* E_0 E_0

Figura 7: Documento sulla valutazione dell'*equity*: valori modificabili nel testo (in grigio) con risultato calcolato (in giallo), equazioni dei modelli FCFE ed EVA.

5.6 Il documento come sorgente

Ogni documento è un file di testo in elm-markup, leggibile e modificabile anche fuori dall'applicazione (Figura 8; un estratto nell'appendice B). Metadati, titoli, blocchi Sheet che richiamano i dataset per nome e blocchi Equation in LaTeX rendono il documento versionabile con gli strumenti del software.

Markup Editor

```

1 |> Metadata
2   author = Luca Erzegovesi
3   title = Testing mathlive /foldBook/
4   description =
5     Test page for testing /foldBook/ with mathlive compute engine.
6
7 |> Title
8   Fogli di calcolo di prova
9
10  Provvisoriamente alla fine di `XModel.elm`
11
12 |> Sheet
13   datasetRef = Az
14   title = Aziende
15
16
17 |> Sheet
18   datasetRef = Ce
19   title = Conto economico
20
21
22 |> Sheet
23   datasetRef = Macro
24   title = Scenario
25
26
27 |> Title
28   Equazioni in formato /LaTeX/ con MathLive
29
30 |> Subtitle
31   Custom functions and variables
    
```

Figura 8: Il sorgente elm-markup del documento di dimostrazione nell'editor integrato.

5.7 Usi previsti

Il prototipo è stato concepito per due usi. Nella didattica: dispense in cui la teoria (testo ed equazioni) e il modello numerico stanno nella stessa pagina, e in cui lo studente può cambiare le ipotesi e vederne gli effetti. Nella ricerca: sperimentare modelli di pianificazione multidimensionali e la loro documentazione riproducibile. Nella fase pilota il software è rimasto uno strumento di sperimentazione dell'autore. Questi obiettivi sono stati ripresi da Impromptu, che è stato progettato per arrivare all'uso effettivo.

6 Limiti e sviluppi: da elm-foldbook a Impromptu

6.1 Limiti del prototipo

La versione 0.7 ha raggiunto gli obiettivi di verifica, ma ha limiti che ne impediscono l'uso oltre la sperimentazione:

- **Ordine di calcolo.** Le formule sono valutate nell'ordine di dichiarazione e le dipendenze fra dataset sono dichiarate a mano. Manca un grafo delle dipendenze, e i riferimenti circolari (frequentissimi nei modelli finanziari, per esempio interessi e debito) non sono gestiti.
- **Prestazioni.** Interpretare Elm nel browser è adatto a modelli piccoli, non a modelli con centinaia di migliaia di celle.
- **Persistenza.** I modelli sono definiti nel codice. Non esiste un formato di file per salvarli, e si salvano solo i documenti.
- **Due motori separati.** Equazioni e fogli convivono nello stesso documento ma sono valutati da motori diversi (Compute Engine in JavaScript e interprete Elm) e non condividono le variabili.
- **Robustezza.** Alcuni esempi dei documenti di prova sono sperimentali e restituiscono valori errati. La gestione degli errori nelle formule è ancora rudimentale.

6.2 Lezioni apprese e decisione architeturale

Il prototipo ha confermato il valore di tre scelte: il modello a dimensioni condivise, le formule a livello di array e la griglia pivot con vassoi. Ha anche mostrato che il calcolo non deve stare nel browser. Serve un motore con un vero grafo delle dipendenze, prestazioni adeguate a modelli reali e un unico linguaggio per le formule dei fogli e le espressioni nei documenti. Da qui la decisione di separare l'interfaccia, che resta in Elm, dal motore, che diventa un server.

6.3 Impromptu

Impromptu, sviluppato dall'autore dal febbraio 2026, è un foglio di calcolo multidimensionale *nella tradizione di Lotus Improv* con un motore di calcolo in Julia (Bezanson et al. 2017) e un client Elm (Erzegovesi 2026). La continuità con elm-foldbook è diretta e verificabile:

- il primo commit del front end Elm di Impromptu (22/02/2026) contiene già, trapiantati e riorganizzati, la griglia pivot (SpreadsheetUI), i vassoi (DnDTray), le viste (XView) e i tipi del modello (XModel, Dataset, DataArray, DimOrDAR, Tray, HeaderTree) di elm-foldbook;
- l'interprete Elm e CalcEngine sono stati sostituiti da un server Julia, che mantiene il modello autorevole come array con dimensioni etichettate (DimensionalData.jl) e comunica con il client via HTTP e WebSocket;
- il linguaggio di formula è a livello di array e viene compilato in Julia. Ha riferimenti fra dataset, filtri per coordinata e operatori posizionali (anno precedente, successivo, primo, ultimo);
- il ricalcolo segue l'ordine topologico delle dipendenze, e i cicli vengono risolti per convergenza;
- i documenti diventano report Markdown che interpolano espressioni del modello: formule e testo usano finalmente lo stesso motore. Si aggiungono grafici Vega-Lite, persistenza su file e database, lavoro multiutente e un'interfaccia per agenti di intelligenza artificiale (Model Context Protocol).

Tabella 3: elm-foldbook 0.7 e Impromptu a confronto.

	elm-foldbook 0.7 (2024)	Impromptu (2026)
Architettura	tutto nel browser	client Elm + server Julia
Modello dati	XModel (array piatti, strides)	DimensionalData.jl
Formule	moduli Elm interpretati	linguaggio ad array compilato in Julia
Ricalcolo	ordine di dichiarazione	ordine topologico, cicli per convergenza
Griglia pivot	SpreadsheetUI, DnDTray	evoluzione degli stessi moduli, finestre lato server
Documenti	elm-markup + MathLive	Markdown con espressioni del modello, KaTeX
Persistenza	solo documenti	file JSON, PostgreSQL
Collaborazione	–	multiutente, annulla/ripeti
Disponibilità	pubblico, BSD-3	progetto di ricerca privato

6.4 Stato attuale e diffusione

Impromptu è un progetto di ricerca personale, non ancora pubblico, condiviso con una cerchia ristretta di sviluppatori e utenti esperti. Architettura e funzionalità sono descritte sul blog *The Wor-*

kbench (<https://theworkbench.lerzegov.org>), in particolare nel post *Impromptu under the hood*, e in video sul canale YouTube <https://www.youtube.com/@TheWorkbench.lerzegov>. In questo percorso *elm-foldbook* è il primo stadio documentato e pubblicamente verificabile.

7 Disponibilità, licenza e riproducibilità

7.1 Dove si trova

- Repository: <https://github.com/lerzegov/elm-foldbook>. La versione di riferimento è il tag `v0.7`; i rilasci pubblici sono i tag `v0.7.1` e `v0.7.2`.
- Motore di calcolo (sviluppo originale): <https://github.com/lerzegov/elm-interpreter-fork>, pubblico dal 29/07/2024.
- Zenodo, *elm-foldbook* 0.7 (archivio del tag `v0.7.2`, 30/09/2026): DOI 10.5281/zenodo.23056100; DOI di tutte le versioni 10.5281/zenodo.23056099.
- Zenodo, motore di calcolo (archivio del commit `bc148ee`, ultimo push pubblico del 18/12/2024, depositato il 30/09/2026 senza modifiche): DOI 10.5281/zenodo.23056240.
- Software Heritage. Gli identificativi SWHID derivano dal contenuto: coincidono con gli hash git del tag e del commit e si possono verificare su <https://archive.softwareheritage.org>.
 - versione di riferimento (tag `v0.7`, 16/12/2024):
`swh:1:rel:1a45a295993b1acee16b332b8dfa3aea2ad3d300`
 - motore di calcolo (commit `bc148ee`, 18/12/2024):
`swh:1:rev:bc148ee3cdb33f435fd68f575a508c01444768ea`

7.2 Licenza

elm-foldbook è distribuito con licenza BSD 3-Clause, la stessa di *elm-interpreter*. I componenti di terzi hanno licenze permissive: MathLive, Compute Engine, CodeMirror e i font KaTeX sono sotto MIT; i pacchetti Elm sono sotto BSD-3 o MIT; *speech-rule-engine* è sotto Apache-2.0. Sono elencati con le note di copyright nel file `THIRD_PARTY_NOTICES.md` del repository.

7.3 Eseguire e ricompilare

Il repository contiene l'applicazione compilata. Per eseguirla basta un server web statico:

```
git clone https://github.com/lerzegov/elm-foldbook.git
cd elm-foldbook/public && python3 -m http.server 8000
```

Per ricompilarla dai sorgenti serve solo il compilatore Elm 0.19.1 (`elm make src/Main.elm --output public/ui.js`). Il 29/09/2026 la ricompilazione è stata verificata da un clone pulito. Il bundle MathLive ricompilato con `npm ci` e Rollup risulta identico byte per byte a quello del 2024.

7.4 Nota sulla cronologia e sulla versione 0.7.1

Il software è stato sviluppato fra giugno e dicembre 2024. La versione 0.7 è del 16/12/2024. Il repository dell'applicazione, privato fino al 2026, è stato reso pubblico nel 2026 con il rilascio 0.7.1, che non modifica il codice applicativo e aggiunge licenze e documentazione; il rilascio 0.7.2 aggiorna solo il README e i metadati per Zenodo. Il fork con il motore di calcolo è invece pubblico dal 29/07/2024: il deposito Zenodo del 2026 ne riproduce senza modifiche lo stato dell'ultimo push del 18/12/2024 e porta quella data di pubblicazione.

C'è un solo punto delicato. La v0.7 era stata compilata contro lo stato del fork dell'interprete presente sul disco il 16/12/2024, committato due giorni dopo insieme a un refactoring. Nella 0.7.1 quello stato è stato ricostruito e incluso in `vendor/`. La ricostruzione è stata verificata sul programma compilato conservato nel commit del 16/12/2024: 1.256 definizioni su 1.281 sono identiche byte per byte, e le altre differiscono solo per dettagli non funzionali. La procedura è documentata in `vendor/README.md`.

Ringraziamenti

Il motore di calcolo si fonda su `elm-interpreter` di Leonardo Taglialegne, che l'autore ringrazia anche per il confronto durante lo sviluppo. Un ringraziamento va anche agli autori dei componenti aperti su cui il progetto si appoggia: Matthew Griffith (`elm-markup`, `elm-ui`), integral424 (`elm-pivot-table`, su cui sono state prototipate le intestazioni gerarchiche), annaghi (`dnd-list`), Arno Gourdol (`MathLive` e `Compute Engine`), Marijn Haverbeke (`CodeMirror`).

Uso di assistenti di intelligenza artificiale. Alcune parti del codice del 2024 sono state scritte con l'aiuto di ChatGPT, come segnalato nei commenti del sorgente. La preparazione del rilascio pubblico 0.7.1 e la stesura di questa nota sono state assistite da Claude (Anthropic). Scelte progettuali, contenuti e verifiche sono dell'autore.

Riferimenti bibliografici

- Bezanson, Jeff, Alan Edelman, Stefan Karpinski, e Viral B. Shah. 2017. «Julia: A Fresh Approach to Numerical Computing». *SIAM Review* 59 (1): 65–98.
- Codd, E. F., S. B. Codd, e C. T. Salley. 1993. *Providing OLAP to User-Analysts: An IT Mandate*. E. F. Codd & Associates.
- Czaplicki, Evan, e Stephen Chong. 2013. «Asynchronous Functional Reactive Programming for GUIs». *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, 411–22.
- Erzegovesi, Luca. 2026. «Impromptu under the hood». The Workbench. <https://theworkbench.lerzegov.org/posts/impromptu/>.
- Gordon, Andy, Simon Peyton Jones, et al. 2021. «LAMBDA: The Ultimate Excel Worksheet Function». Microsoft Research Blog.
- Gourdol, Arno. s.d. *MathLive and Cortex Compute Engine*. <https://cortexjs.io>.
- Griffith, Matthew. 2018. *elm-markup*. Released. <https://github.com/mdgriffith/elm-markup>.
- Hermans, Felienne, Martin Pinzger, e Arie van Deursen. 2015. «Detecting and Refactoring Code Smells in Spreadsheet Formulas». *Empirical Software Engineering* 20 (2): 549–75.
- Hoyer, Stephan, e Joe Hamman. 2017. «xarray: N-D Labeled Arrays and Datasets in Python». *Journal of Open Research Software* 5 (1): 10.
- Kluyver, Thomas et al. 2016. «Jupyter Notebooks: A Publishing Format for Reproducible Computational Workflows». In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. IOS Press.

- Knuth, Donald E. 1984. «Literate Programming». *The Computer Journal* 27 (2): 97–111.
- McKinney, Wes. 2010. «Data Structures for Statistical Computing in Python». *Proceedings of the 9th Python in Science Conference*, 56–61.
- Panko, Raymond R. 1998. «What We Know About Spreadsheet Errors». *Journal of End User Computing* 10 (2): 15–21.
- Peyton Jones, Simon, Alan Blackwell, e Margaret Burnett. 2003. «A User-Centred Approach to Functions in Excel». *Proceedings of the Eighth ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 165–76.
- Quantrix. s.d. *Quantrix Modeler*. <https://www.quantrix.com>.
- Taglialegne, Leonardo. 2023. *elm-interpreter*. Released. <https://github.com/miniBill/elm-interpreter>.

A Moduli del software

LE = L. Erzegovesi, LT = L. Taglialegne. Righe alla revisione di dicembre 2024.

Tabella 4: Moduli principali.

Modulo	Righe	Origine	Funzione
<i>Applicazione (src/, srcjs/)</i>			
Main	507	LE	Applicazione, routing, ciclo di ricalcolo
CalcEngine	1.343	LE	Ambienti dell’interprete, valutazione e scrittura dei risultati
SpreadsheetUI	3.487	LE	Griglia pivot, intestazioni gerarchiche, editing
XView	947	LE	Viste: vassoi, alberi delle intestazioni
DnDTray	456	LE	Vassoi con trascinamento delle dimensioni
DatasetPage	900	LE	Pagina dataset: foglio, editor delle formule
MarkupFoldbook, TypesMarkup	2.963	LE	Documenti elm-markup, blocchi Sheet ed Equation
Altri moduli Elm	1.245	LE	Home, rotte, porte, colori, widget, prove
mathlive-element.ts e config.	620	LE	Componente MathLive, Compute Engine, macro
CodeMirror (recuperato dal bundle)	309	LE	Editor di formule e di documenti; linguaggio elm-markup
<i>Motore (vendor/elm-interpreter-fork/src)</i>			
XModel	4.145	LE	Modello multidimensionale, selezione, editing strutturale, allineamento
TypesXModel	957	LE	Tipi del modello, vassoi, intestazioni
XModelSample (poi in XModel)	624	LE	Modello di esempio finPlan
Financial	116	LE	Funzioni finanziarie

Modulo	Righe	Origine	Funzione
XParser, FormulaParser, AppUtil	429	LE	Parsing di celle e nomi, utilità
Kernel	869	LT + LE	Primitive; LE: estensione ad array
Eval/Expression, Eval/Module	1.727	LT + LE	Valutatore; LE: nomi di intervallo, ambienti per dataset
Types, Value, Environment e altri	1.878	LT + LE	Tipi dei valori (DataAr), ambiente
UI/Source, TopologicalSort e altri	1.090	LT	Interprete originale
generated/	–	LT	elm/core rigenerata per l'interprete

B Esempi di sorgente

Modulo delle formule del dataset Macro

```
module Macro exposing (..)
macro__inflazione_forecast = macro__inflazione_base * 1.02
macro__cambioUsdEur_forecast = macro__cambioUsdEur_base * 3.07
macro__cambioCalc = macro__cambioUsdEur * 5
dependentDatasetRefs = ["Ce"]
```

La prima formula calcola l'inflazione dello scenario forecast per tutti gli anni a partire dallo scenario base. L'ultima dichiara che il conto economico dipende da questo dataset e va ricalcolato dopo di esso.

Estratto del documento di dimostrazione (Mathlive.emu)

```
|> Metadata
  author = Luca Erzegovesi
  title = Testing mathlive /foldBook/

|> Title
  Fogli di calcolo di prova

|> Sheet
  datasetRef = Ce
  title = Conto economico

|> Subtitle
  Custom functions and variables

|> Equation
  \operatorname{Psum} \coloneqq \sum_{j=1}^n \frac{c}{(1+y)^j}
  + \frac{v}{(1+y)^n}
```