

D2

aka Mermaid on steroids

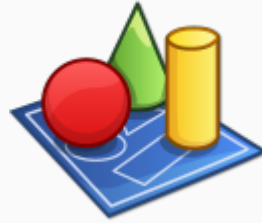
2024-12-19



- 1. État de l’art
- 2. D2
- 3. TALA
- 4. Conclusion

1. État de l'art

1. État de l'art



1. État de l'art — 1.1 WYSIWYG

GUI classique avec disposition manuelle des éléments

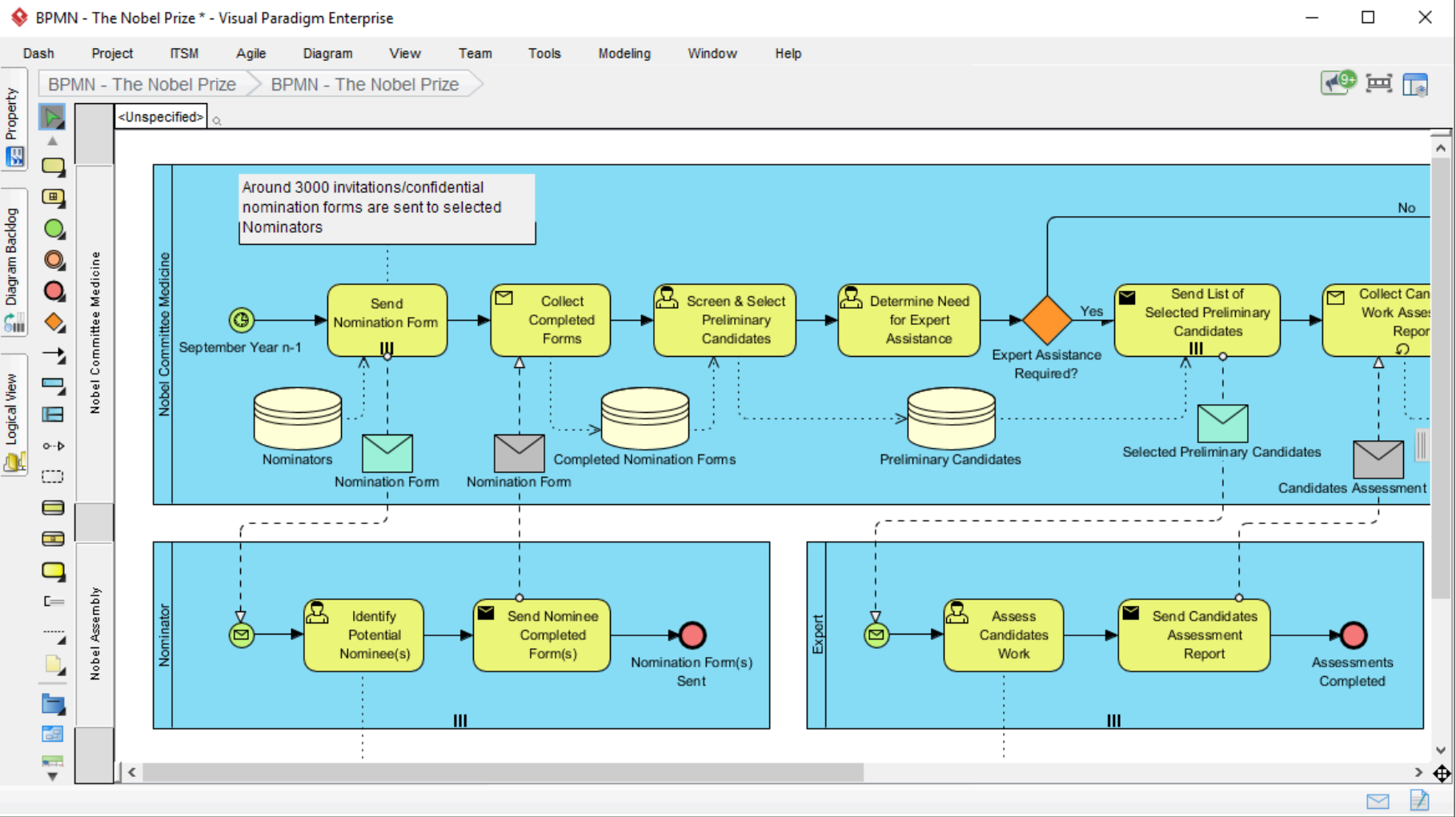
- utilisation intensive de la souris (cliquer-glisser)

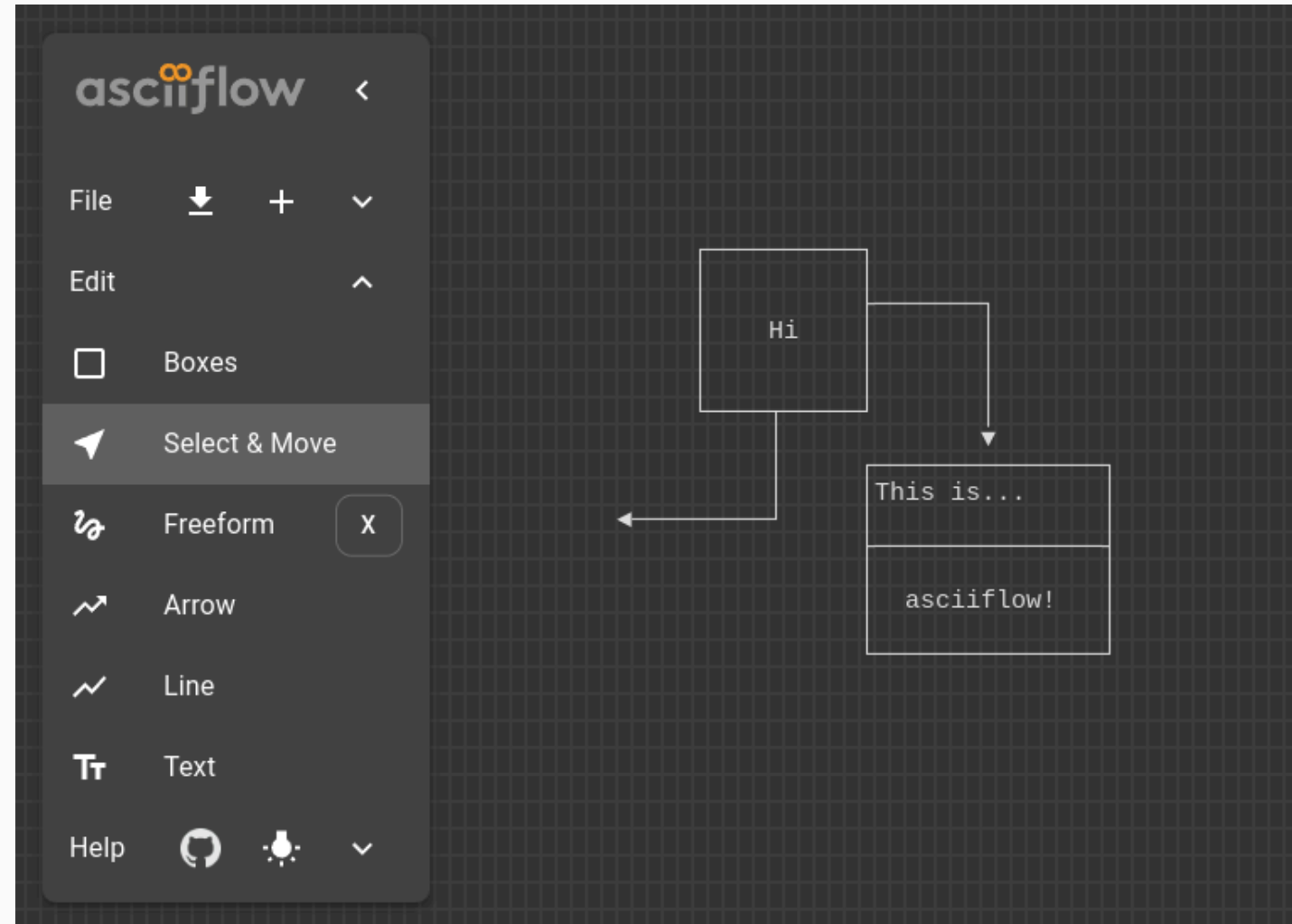
Exemples:

- Enterprise Architect
- Powerdesigner
- Visual Paradigm
- Excalidraw
- MS Visio
- yEd

1.1 WYSIWYG

1. État de l'art — 1.1 WYSIWYG





1. État de l'art — 1.1 WYSIWYG

- On commence à entrer au royaume des nerds
- Placement toujours manuel mais export au bête format .txt
- Approche utilisée dans les RFCs de l'IETF



1. État de l'art — 1.2 Langage déclaratif

aka *Diagram as code*

Objectifs:

- décrire l'ensemble du diag au format texte
- placement automatique des éléments (layout engine)

Exemples:

- GraphViz (1991)
- PlantUML (2009)
- MermaidJS (2014)
- D2 (2022)

1.2 Langage déclaratif

Diagramme de flux avec Mermaid:

flowchart TD

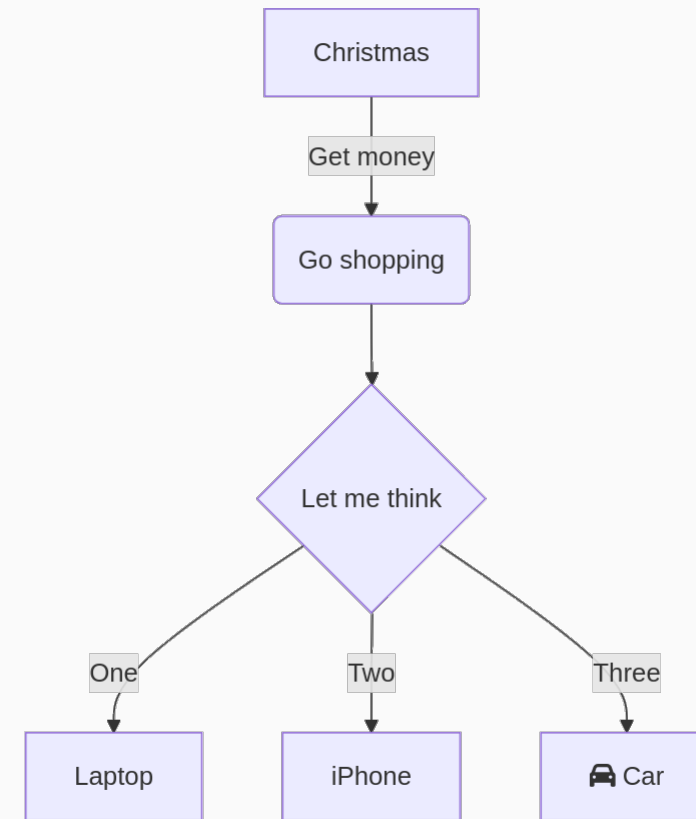
```
Christmas -->|Get money| B(Go shopping)
```

```
B --> C{Let me think}
```

```
C -->|One| D[Laptop]
```

```
C -->|Two| E[iPhone]
```

```
C -->|Three| F[fa:fa-car Car]
```



1. État de l'art — 1.2 Langage déclaratif

WYSIWYG

Langage déclaratif

1. État de l'art — 1.3 Que choisir?

- On reparlera plus tard des moteurs de layout

Distraction:

- *WYSIWYG* – le design du diagramme prend le pas sur le design du système
- *Déclaratif* – un dev peut se « perdre » dans le langage

1.3 Que choisir?

WYSIWYG

- ✓ layout précis et soigné
- ✓ facile à prendre en main
- ✓ accessible à tous·tes

Langage déclaratif

- ✓ layout auto-généré
- ✓ facilement versionnable
- ✓ maintenance aisée
- ✓ motivant

1. État de l'art — 1.3 Que choisir?

- On reparlera plus tard des moteurs de layout

Distraction:

- *WYSIWYG* – le design du diagramme prend le pas sur le design du système
- *Déclaratif* – un dev peut se « perdre » dans le langage

1.3 Que choisir?

WYSIWYG

- ✓ layout précis et soigné
- ✓ facile à prendre en main
- ✓ accessible à tous·tes
- x chronophage
- x modification coûteuse
- x complexité du format de données
- x source de distraction

Langage déclaratif

- ✓ layout auto-généré
- ✓ facilement versionnable
- ✓ maintenance aisée
- ✓ motivant
- x médiocrité des moteurs de layout
- x offre réduite et peu maintenue
- x parfois source de distraction

1. État de l'art — 1.3 Que choisir?

- On reparlera plus tard des moteurs de layout

Distraction:

- **WYSIWYG** – le design du diagramme prend le pas sur le design du système
- **Déclaratif** – un dev peut se « perdre » dans le langage

1.3 Que choisir ?

Historiquement:

- faible complexité et/ou esthétique secondaire → **déclaratif**
- forte complexité et/ou esthétique soignée → **WYSIWYG**

1. État de l'art — 1.3 Que choisir ?

- diagramme de séquence paiement CB → déclaratif
- schéma d'architecture APPP présenté lors d'un audit → WYSIWYG

Il n'y a pas de mauvaise approche, que des mauvais choix.

2. D2



2. D2

—

- *Declarative Diagramming*
- rendu public en 2022
- codé en Go
- langage déclaratif du même nom



Logo de D2

Lisibilité > Concision

Mermaid:

```
A[(Christmas)]
```

2. D2

— 2.2 Choix de conception

- D2 est verbeux au profit d'une meilleure lisibilité
- Petit jeu : que va rendre ce bout de code Mermaid ?
- Mermaid est très concis mais parfois cryptique...

Lisibilité > Concision

Mermaid:

```
A[(Christmas)]
```

D2:

```
A: Christmas {shape: cylinder}
```

2. D2

— 2.2 Choix de conception

- D2 est verbeux au profit d'une meilleure lisibilité
- Petit jeu : que va rendre ce bout de code Mermaid ?
- Mermaid est très concis mais parfois cryptique...

Design du système > Design du diagramme

- Joli par défaut
 - *opinionated*
- Séparation claire entre éléments du diagramme et style
 - classes à la HTML/CSS
 - fichier de style séparé

— 2.2 Choix de conception

- Inutile de configurer 1000 choses pour avoir un beau diagramme
 - inconvénient: D2 a sa propre idée de ce que «beau» signifie (*opinionated*)

2.3 Exemple

christmas.d2

...@styles

Christmas.class: [emphasis; hohoho]

Go: Go shopping

LMT: Let me think {class: alt}

Car.icon: https://...car.png

Christmas -> Go: Get money

Go -> LMT

LMT -> Laptop: One

LMT -> iPhone: Two

LMT -> Car: Three

styles.d2

```
classes: {  
  hohoho: {  
    style: {  
      fill: "linear-gradient(#2f7336,  
#aa3a38)"  
    }  
  }  
}
```

```
alt: {  
  shape: diamond  
  style: {  
    stroke: "#f4a261"  
  }  
}  
}
```

2. D2

— 2.3 Exemple

- Ligne de styles omises pour l'exemple
- La déclaration du diagramme reste lisible

2.3 Exemple



2. D2

— 2.3 Exemple

2.4 Applications possibles

- Thèmes (ei.d2, lyf.d2, etc.)
- Séparation modèle / vue
- etc.

2. D2

— 2.4 Applications possibles

- diagrammes réutilisables dans différents contextes
- gain de temps
- favorise le partage de la documentation

2.5 De nombreuses fonctionnalités

- Composition / Animation
- Support blocs de code et Markdown
- Diagrammes spéciaux (ERD, classes, séquence, grille)

À découvrir sur d2lang.com/tour

2. D2

— 2.5 De nombreuses fonctionnalités

- Composition: publier PDF navigable à partir des diagrammes pour présenter ensemble d'un système
- Animation: mouvement lignes pointillés, alternance entre N scénarii dans un seul digramme, etc.

The lack of ability to spacially architect a system should not block the creation of valuable documentation.

2. D2

— 2.5 De nombreuses fonctionnalités

- Terrastruct aurait pu se contenter de proposer une alternative au moins aussi bien que Mermaid
- Leur modèle économique *open core* s'appuie en partie sur un *layout engine* révolutionnaire

3. TALA



3. TALA

—

3.1 Layout engine?

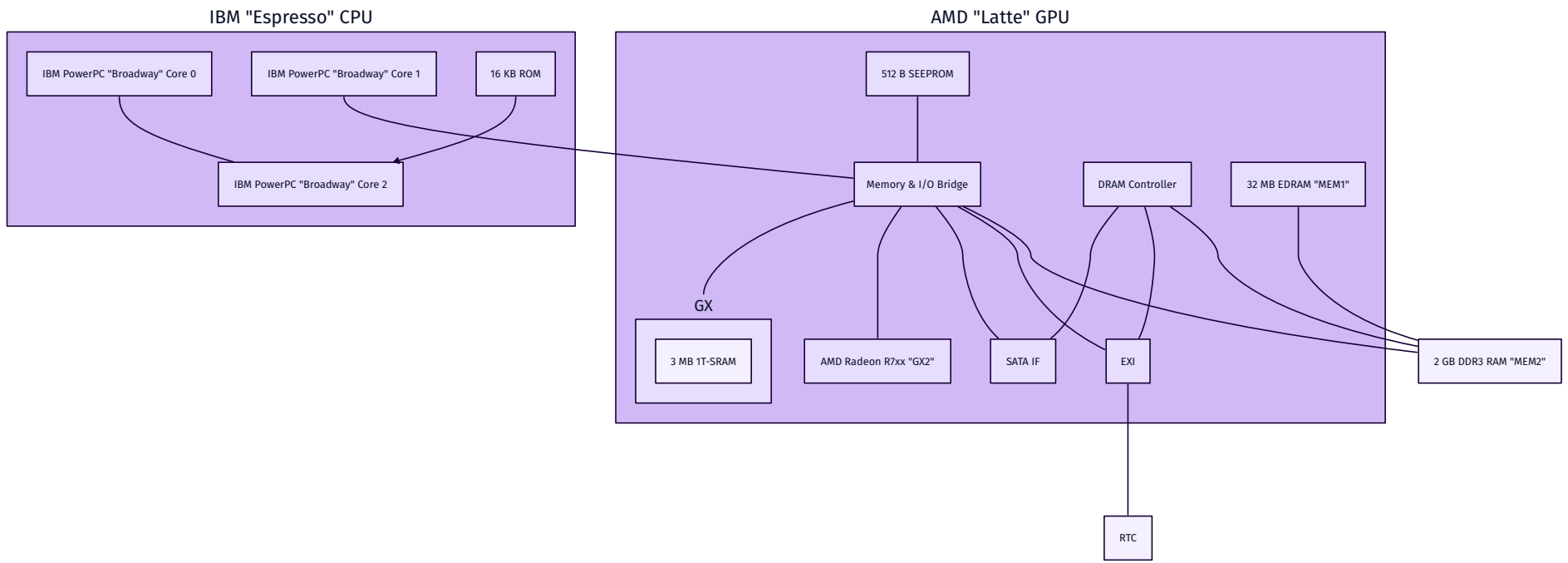
- Logiciel responsable de la bonne disposition spatiale des éléments
- *Nerf de la guerre* de l'approche déclarative
- Problème **très complexe**
- Exemples connus: **Dagre**, **ELK**

3. TALA

— 3.1 Layout engine?

- Grosso modo ce qui permet de lâcher la souris et de gagner du temps
- Des générations de chercheurs en théorie des graphes s'y cassent les dents depuis des décennies
- Même le nouveau moteur développé par Terrastruct n'est pas parfait

3.2 Le problème avec Dagre et ELK

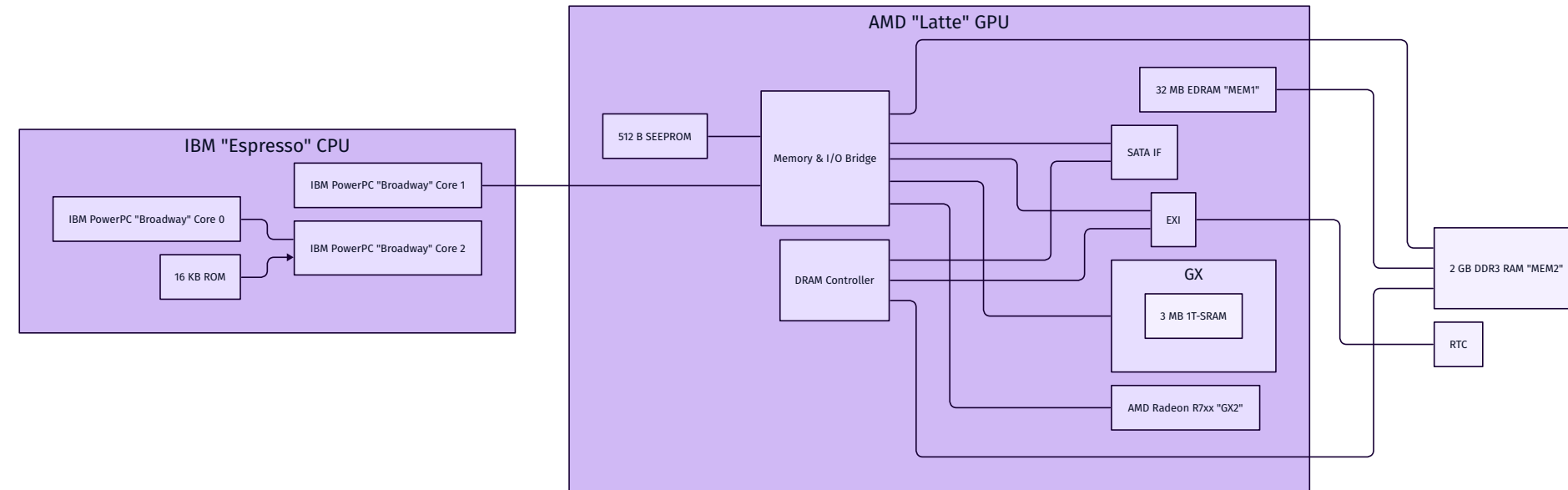


Architecture Wii U avec Dagre

3. TALA

— 3.2 Le problème avec Dagre et ELK

3.2 Le problème avec Dagre et ELK



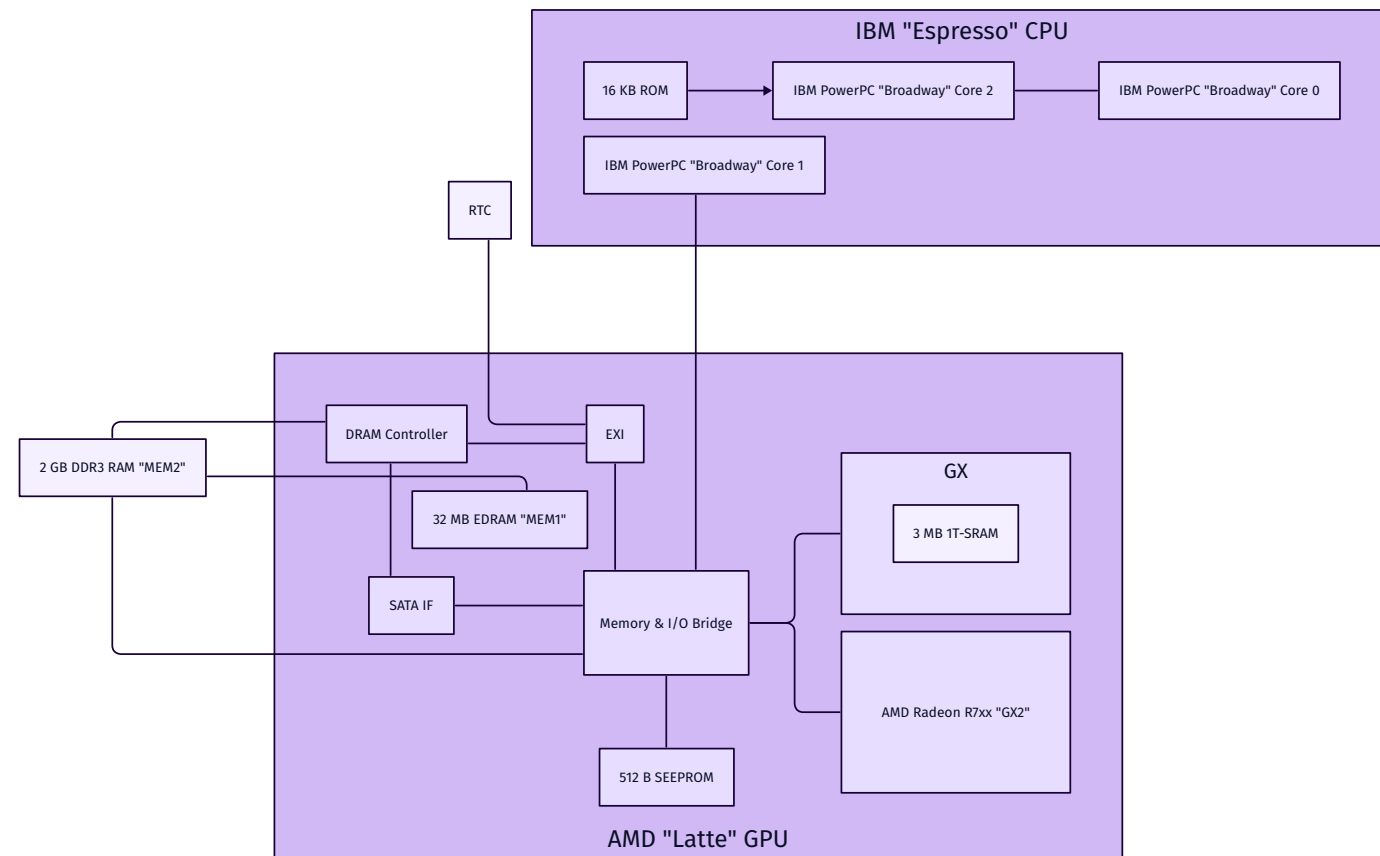
Architecture Wii U avec ELK

3. TALA

— 3.2 Le problème avec Dagre et ELK

- Point commun entre Dagre et ELK: éléments vus de façon hiérarchique
- Marche très bien quand ce qu'on représente est une hiérarchie, un peu moins dans cet exemple

3.3 La nouvelle alternative



Architecture Wii U avec TALA

3. TALA

— 3.3 La nouvelle alternative

- Minimise le nombre de croisements
- Privilégie la symétrie
- Heuristiques qui approcher au maximum ce qu'un humain dessinerait au tableau

Terrastruct's AutoLayout Approach

- Code propriétaire
- Binaire séparé de D2
- Fonctionnalités avancées:
 - Direction par conteneur
 - Contraintes de positionnement
 - *Seed*

Plus d'informations sur terrastruct.com/tala

— 3.3 La nouvelle alternative

- D2 est totalement découplé du moteur de layout sous-jacent
 - Il supporte Dagre, ELK et TALA

Seed:

- grande composante aléatoire dans la génération du placement
- parfois simple changement label = changement de layout radical
- comme dans Minecraft, possible de passer valeur aléatoire pour influencer la génération

4. Conclusion

4. Conclusion

—

- Alternative à LaTeX prometteuse
- Modèle *open core* avec éditeur Web collaboratif
- Communauté active partageant modèles et extensions
- Écrit en Rust 🦀
- Plus d'infos sur typst.app

- LaTeX ou Word, sur la même opposition *code/WYSIWYG*
- Extension utilisée pour écrire cette présentation: Touying

4.2 Meme de circonstance



4. Conclusion

— 4.2 Meme de circonstance

That's all, folks!

4. Conclusion

— 4.2 Meme de circonstance