Extraction des workflows d'applications orientées objets basée sur l'analyse statique du code

SELMADJI Anfel

Co-encadrants : Hinde Lilia Bouziane, Abdelhak-djamel Seriai
Chouki Tibermacine
Directeur de thèse : Christophe Dony

Plan

- Contexte
  - Re-ingénierie d'applications orientées objets
  - Workflows
- Objectif
  - Extraction d'un workflow à partir d'une application orientée objets
- Une approche basée sur l'analyse statique
- Travaux connexes
- Conclusion et perspectives

Plan

- Contexte
  - Re-ingénierie d'applications orientées objets
  - Workflows
- Objectif
  - Extraction d'un workflow à partir d'une application orientée objets
- Une approche basée sur l'analyse statique
  - Travaux connexes
  - Conclusion et perspectives

Re-ingénierie d'applications orientées objets (OO) (1/2)

- Focalisation sur la récupération de la structure
  - Amélioration de la réutilisation et la maintenabilité
- Exemple :

Application OO

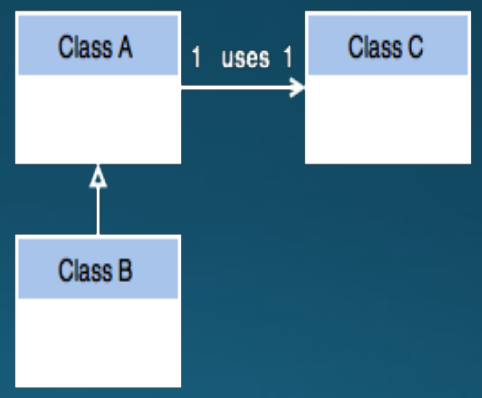
Application à composants

- La structure ne permet pas (ou très peu) d'exprimer la vue temporelle

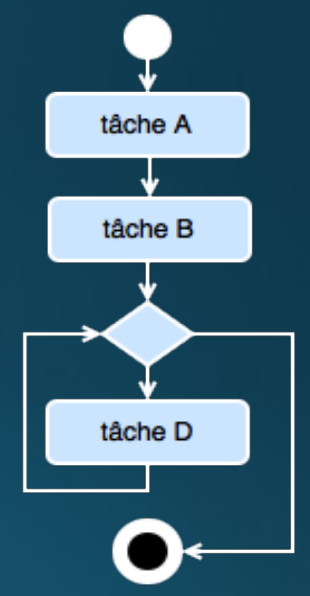
Re-ingénierie d'applications orientées objets (OO)(2/2)

- ***Vue temporelle:*** peut être exprimée par des digrammes d'activités, de séquences, d'états-transitions, des workflows, ...

- Exemples d'utilisation :
  - Compréhension de l'exécution
  - Conception d'applications du commerce
  - Conceptions d'applications de l'e-science
    - Workflow Epigenomics
    - Workflow Montage



Vue structurelle



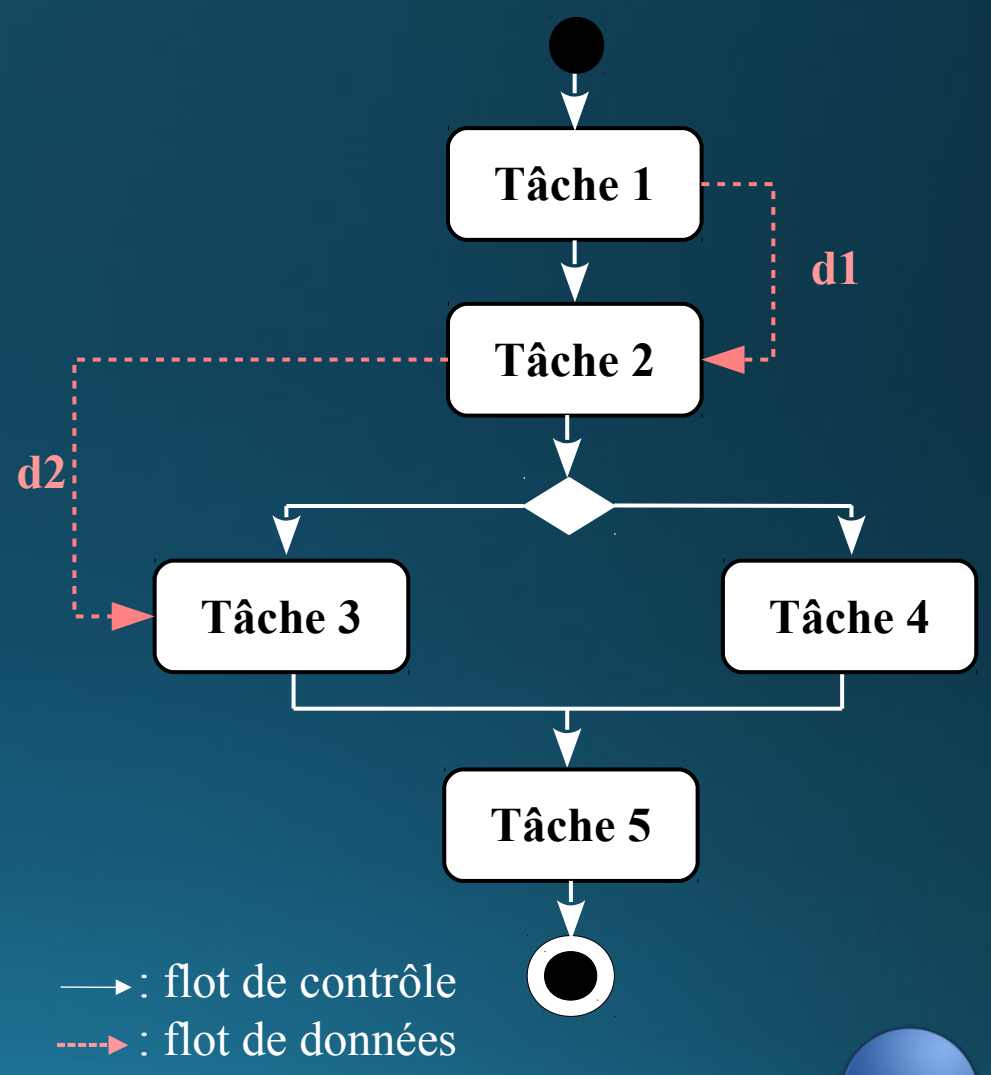
Vue temporelle

Workflow(1/2)

- ***Workflow :*** assemblage décrivant des tâches et des dépendances de données et/ou de contrôle entre ces tâches

- Dépendances de données : flot de données

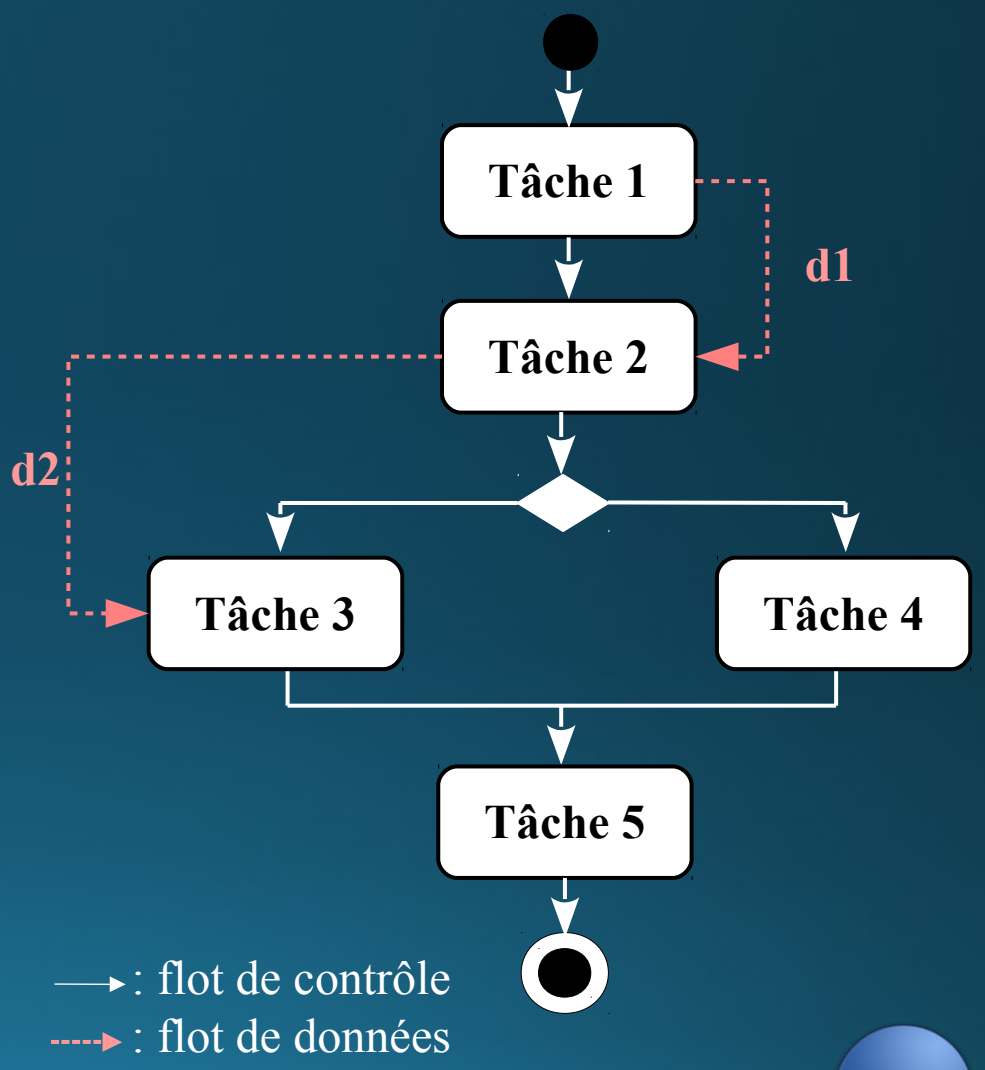
- Dépendances de contrôle : flot de contrôle



Workflow(1/2)

- ***Tâche :***

  - Unité de construction de base d'un workflow
  - Ensemble d'instructions : programme, méthode d'une classe, opération d'un service web,...
  - Définit des données en entrée et des données en sortie



Plan

- Contexte
  - Re-ingénierie d'applications orientées objets
  - Workflows
- Objectif
  - Extraction d'un workflow à partir d'une application orientée objets
- Une approche basée sur l'analyse statique
- Travaux connexes
- Conclusion et perspectives

Objectif

- Extraction d'un workflow à partir d'une application OO

- Principale motivation :
  - Envisager une migration vers une application sous forme de workflow (utilisant des technologies de composants ou de services)

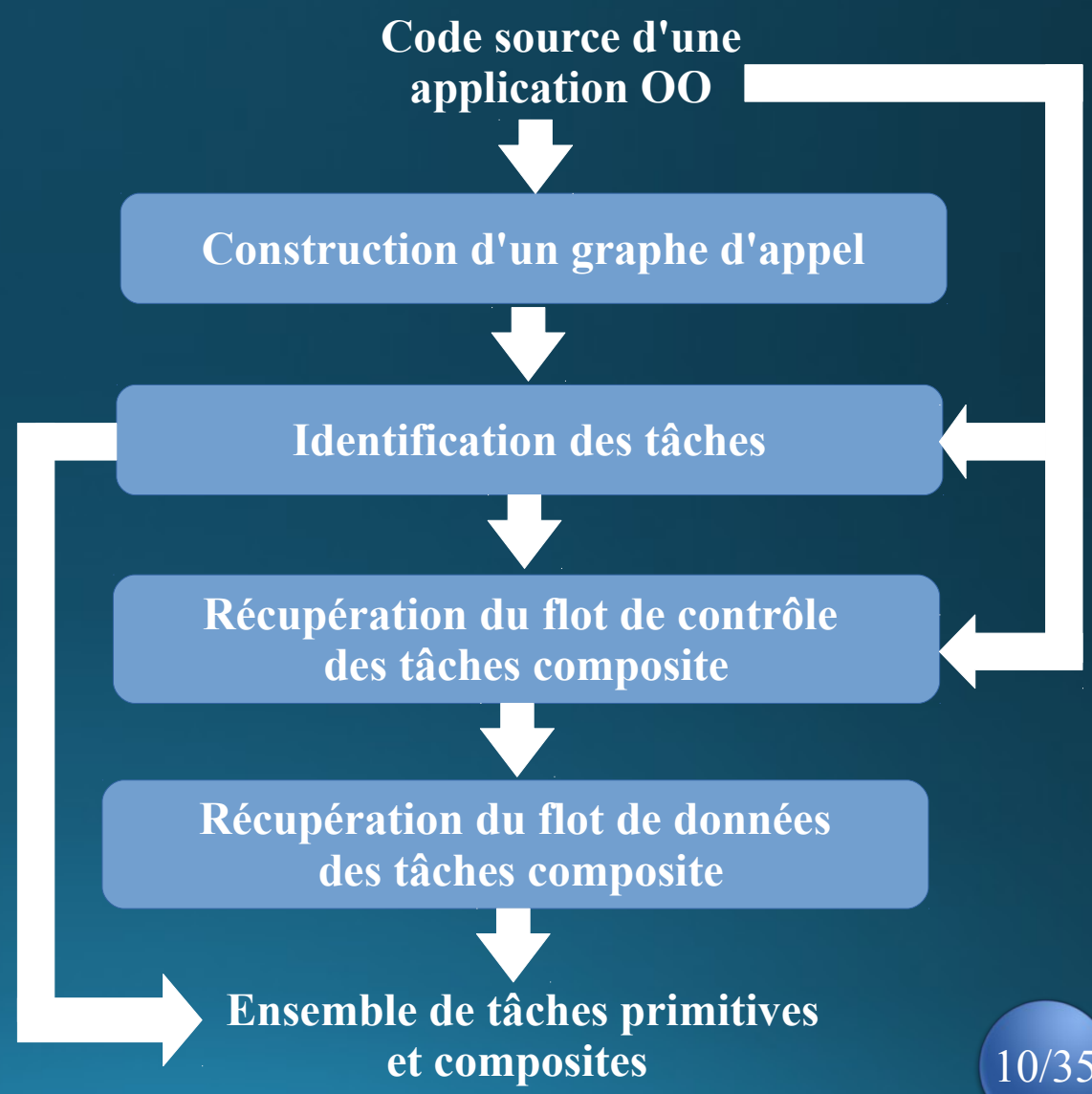
- Principaux avantages
  - Meilleure compréhension de l'exécution
  - Possibilité d'exécution sur le Cloud à moindre coût

Plan

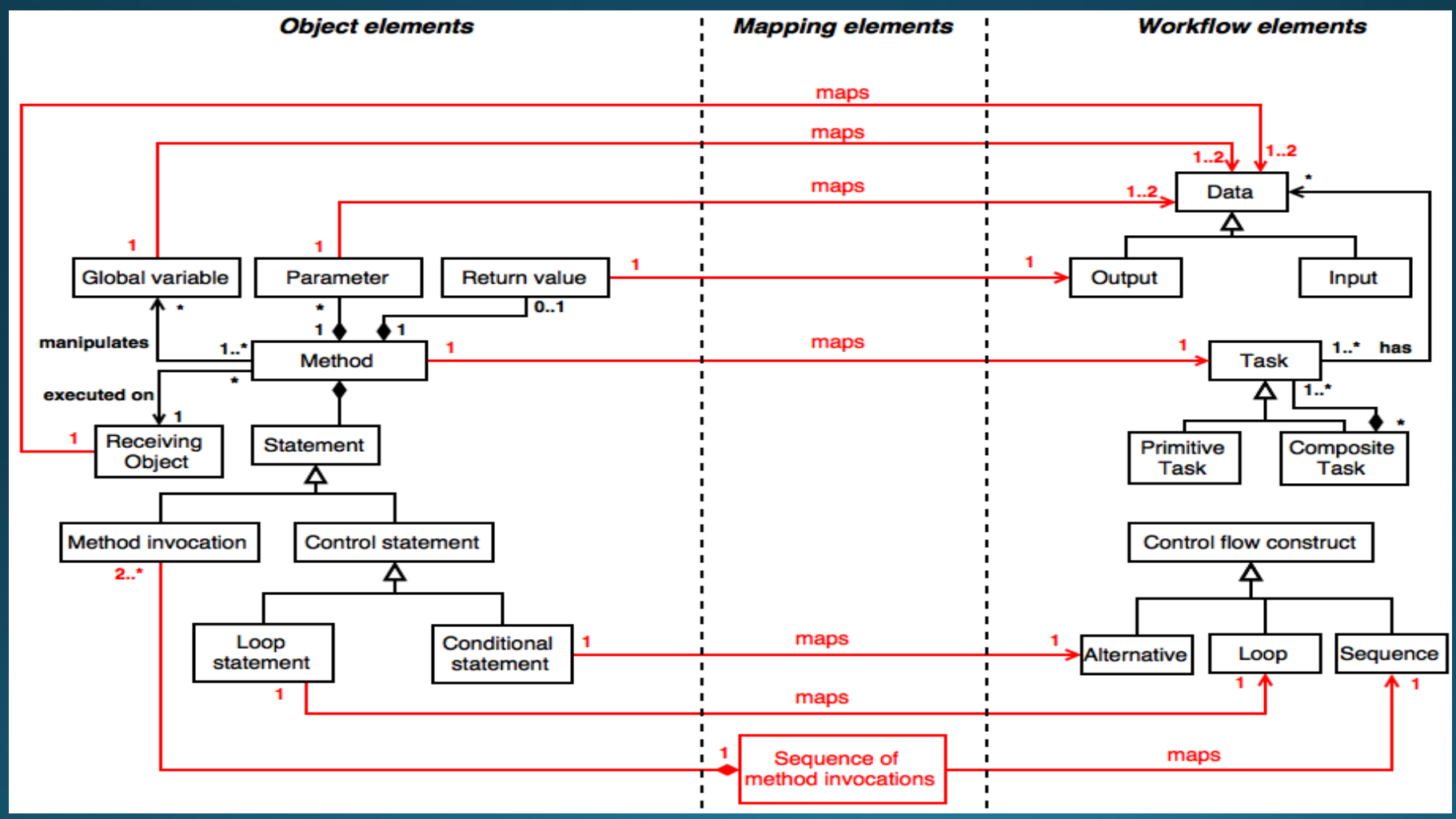
- Contexte
  - Re-ingénierie d'applications orientées objets
  - Workflows
- Objectif
  - Extraction d'un workflow à partir d'une application orientée objets
- Une approche basée sur l'analyse statique
- Travaux connexes
- Conclusion et perspectives

Etapes de l'extractions d'un workflow

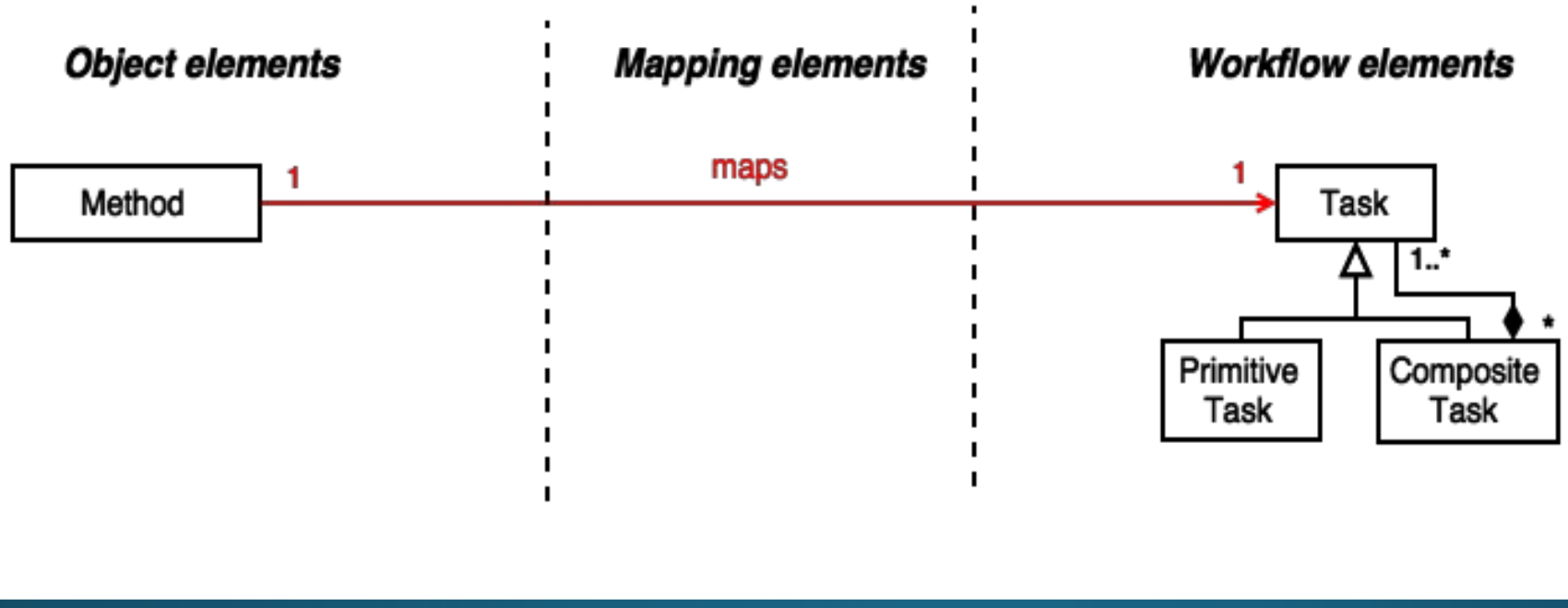
1. Proposition d'un modèle de correspondance qui associe à chaque concepts dans les applications OO son correspondant dans les workflows
2. Analyse pour l'identification des tâches
3. Analyse pour l'identification du flot de contrôle
4. Analyse pour l'identification du flot de données



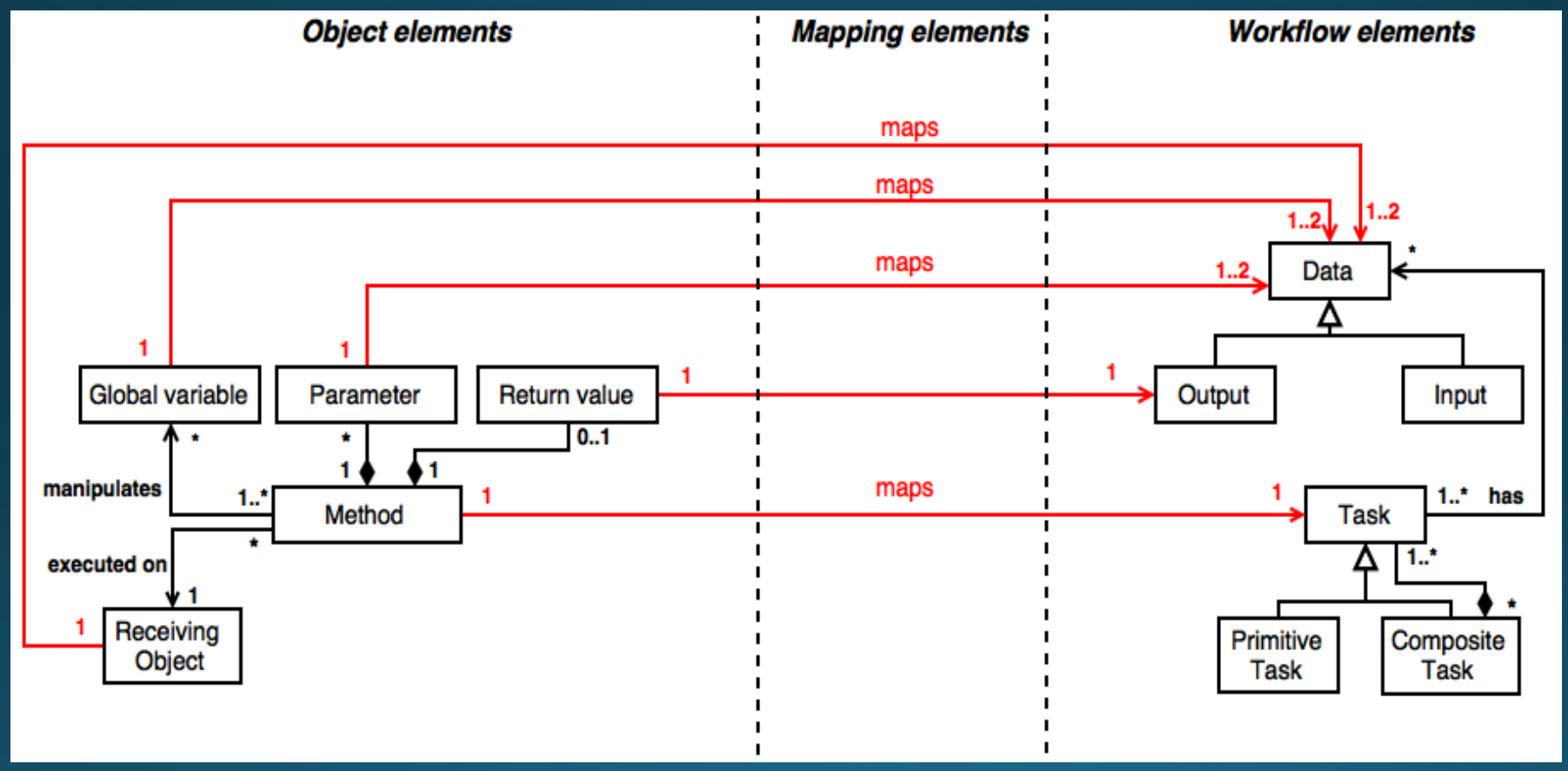
Modèle de correspondance entre les concepts des applications OO et les concepts des workflows



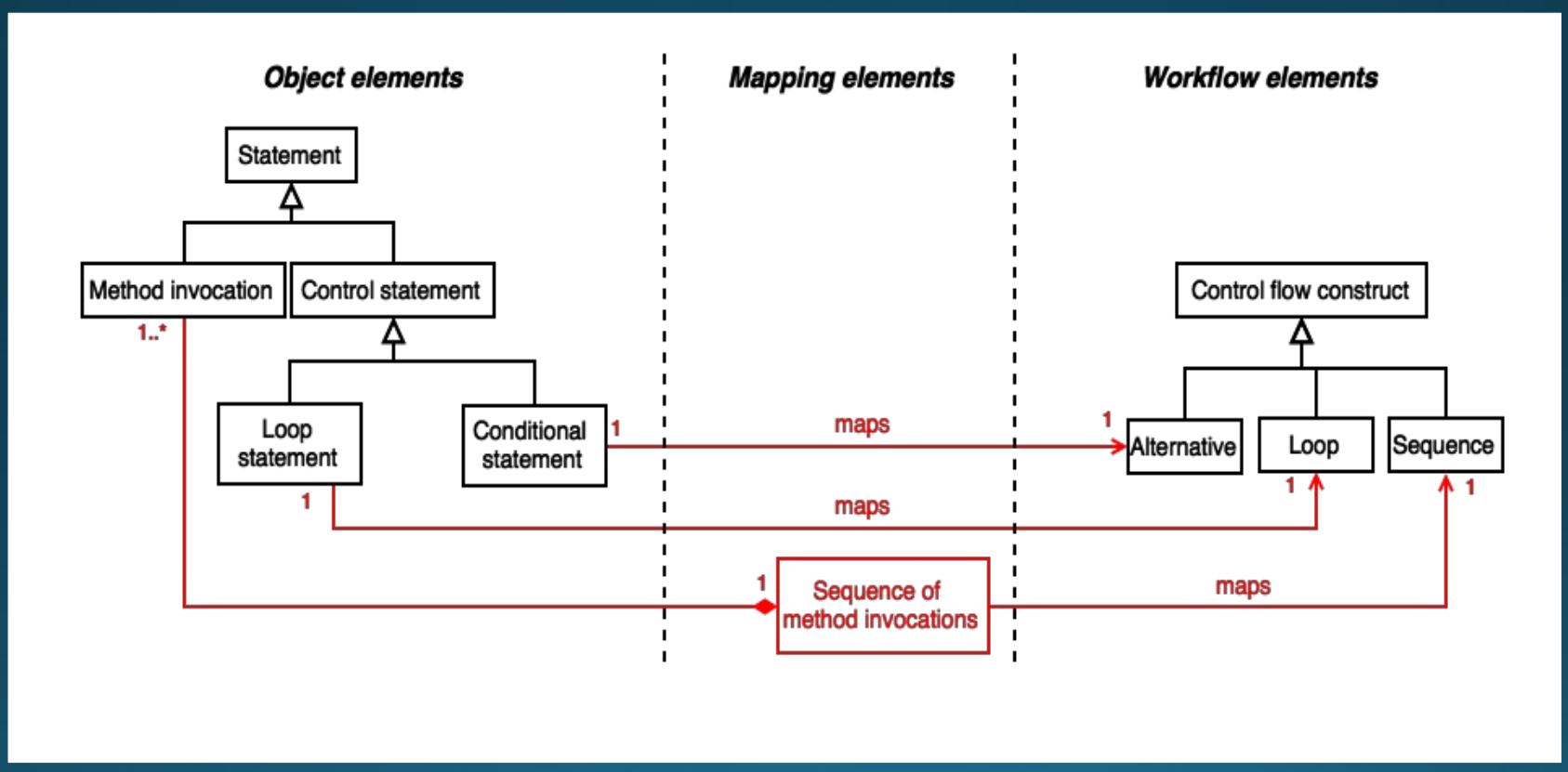
Zoom sur une tâche



Zoom sur le flot de données

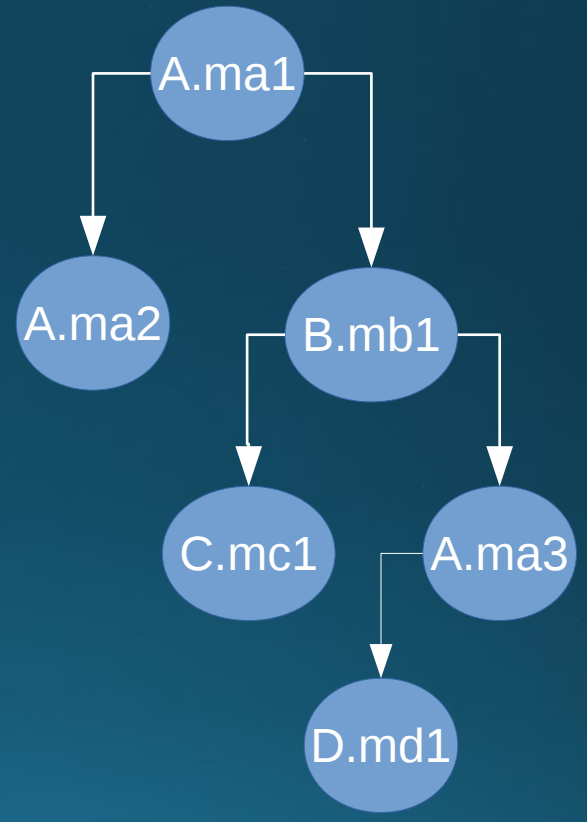


Zoom sur le flot de contrôle



Analyse statique : identification des tâches (1/7)

- Graphe d'appel :
  - Nœuds : méthodes
  - Arcs : appels de méthodes
- Tâche à partir du graphe d'appel : point d'entrée d'un sous-graphe connexe sans arcs sortants
- Point d'entrée : nœud sans arcs entrants



Graphe d'appel

Analyse statique : identification des tâches (2/7)

- **Application Library**

```
public class Subscriber{
 int numSubscriber ;
 ArrayList<Book> borrowedBooks;
 public Subscriber(int number) {
   numSubscriber = number;
   borrowedBooks=new ArrayList<Book>();}
 public int nbBorrowedBooks(){
   return borrowedBooks.size();}
 public void addBook(Book book){
   borrowedBooks.add(book);}
 public void removeBook(Book book){
   borrowedBooks.remove(book);}
 public void borrowBook(Book book){
   if (nbBorrowedBooks()>=5){
      System.out.println("Error");}
   else{
      addBook(book);
      book.borrow(this);}}
public boolean renderBook (Book book){
   boolean returned=true;
   if (! borrowedBooks.contains(book)){
      returned=false;
      System.out.println("Error");}
   else{
      book.render();
      removeBook (book);}
   return  returned;}}
```

Classe Subscriber

```
public class Book{
 String author;
 String title;
 boolean available;
 Subscriber borrower;
 public Book(){}
 public Book(String title, String author) {
   this.title = title;
   this. author = author;
   available= true;}
 public void borrow(Subscriber subscriber){
   if (available){
      available=false;
      borrower=subscriber;}
   else
      System.out.println("Unavailable");}
 public void render(){
   available=true;}}
```

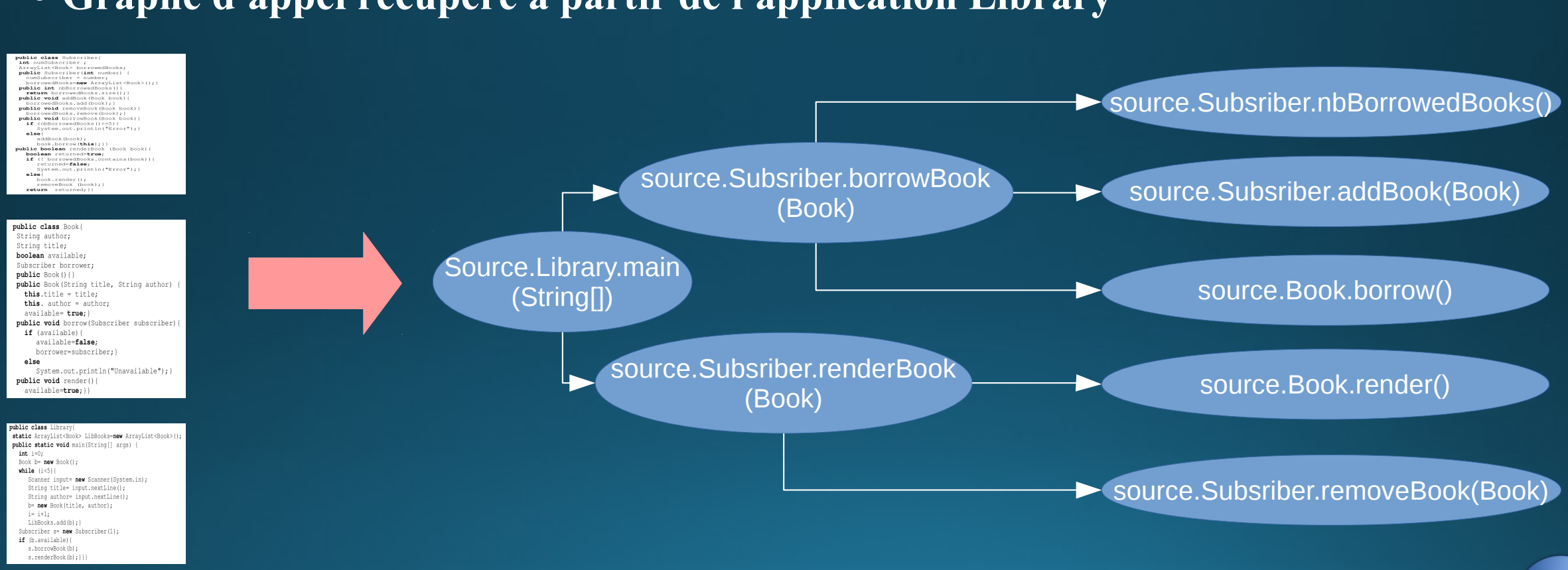
Classe Book

```
public class Library{
 static ArrayList<Book> LibBooks=new ArrayList<Book>();
 public static void main(String[] args) {
   int i=0;
   Book b= new Book();
   while (i<5){
      Scanner input= new Scanner(System.in);
      String title= input.nextLine();
      String author= input.nextLine();
      b= new Book(title, author);
      i= i+1;
      LibBooks.add(b);}
 Subscriber s= new Subscriber(1);
 if (b.available){
      s.borrowBook(b);
      s.renderBook(b);}}}
```

Classe Library

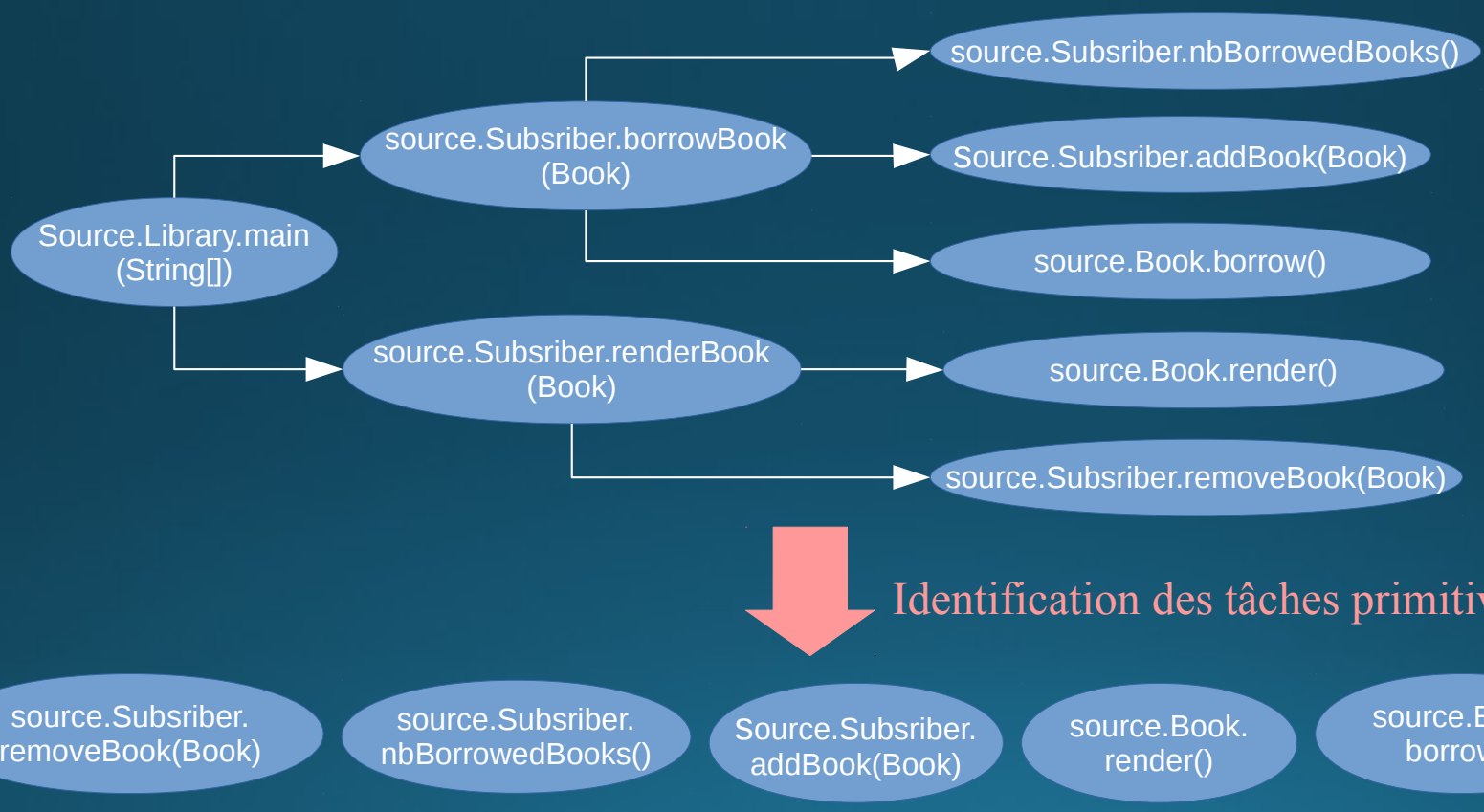
Analyse statique : identification des tâches (3/7)

- **Graphe d'appel récupéré à partir de l'application Library**



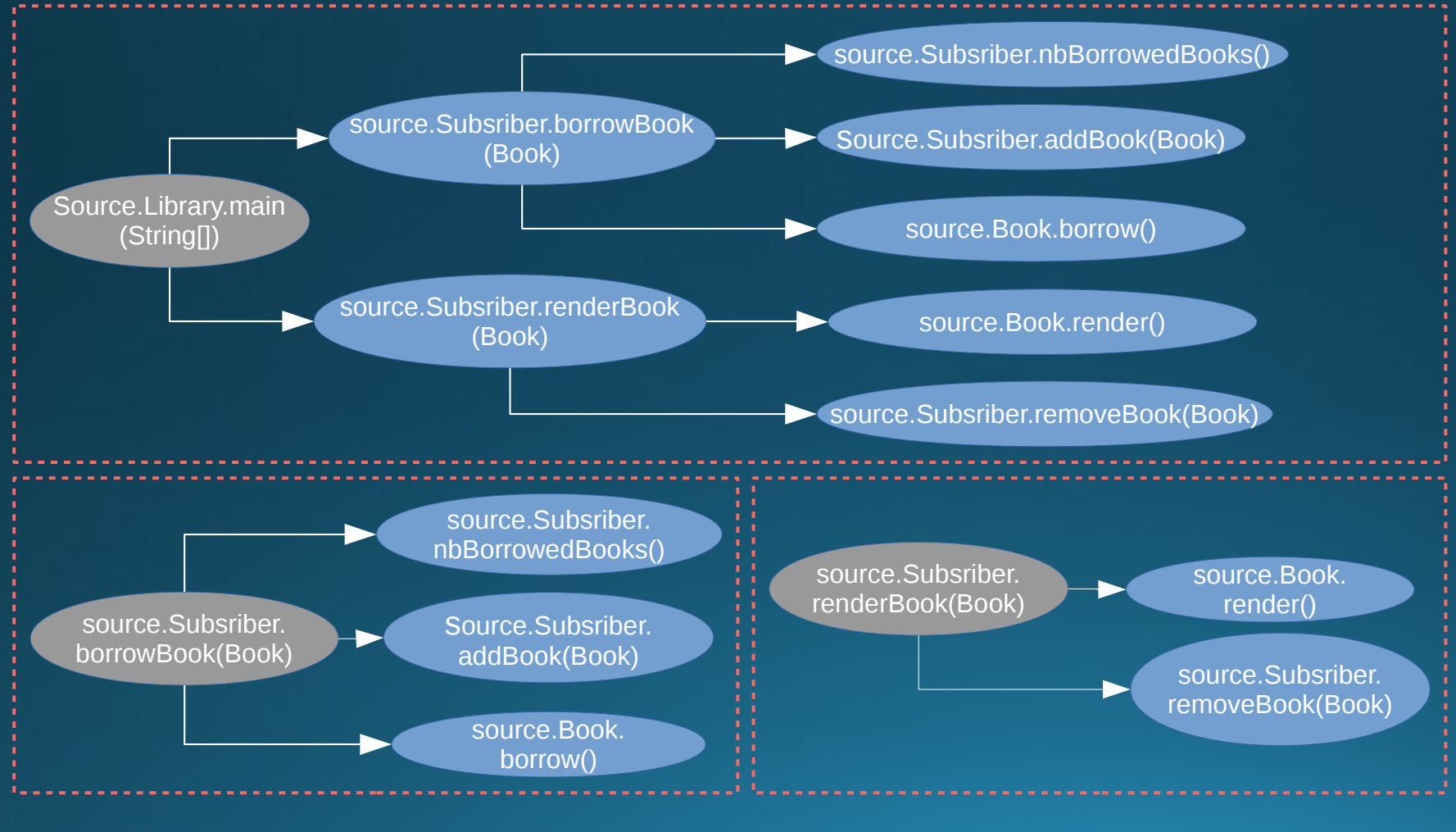
Analyse statique : identification des tâches (4/7)

- **Tâches primitives récupérées à partir du graphe d'appel de l'application Library**



Analyse statique : identification des tâches (5/7)

- **Tâches composites récupérées à partir du graphe d'appel de l'application Library**

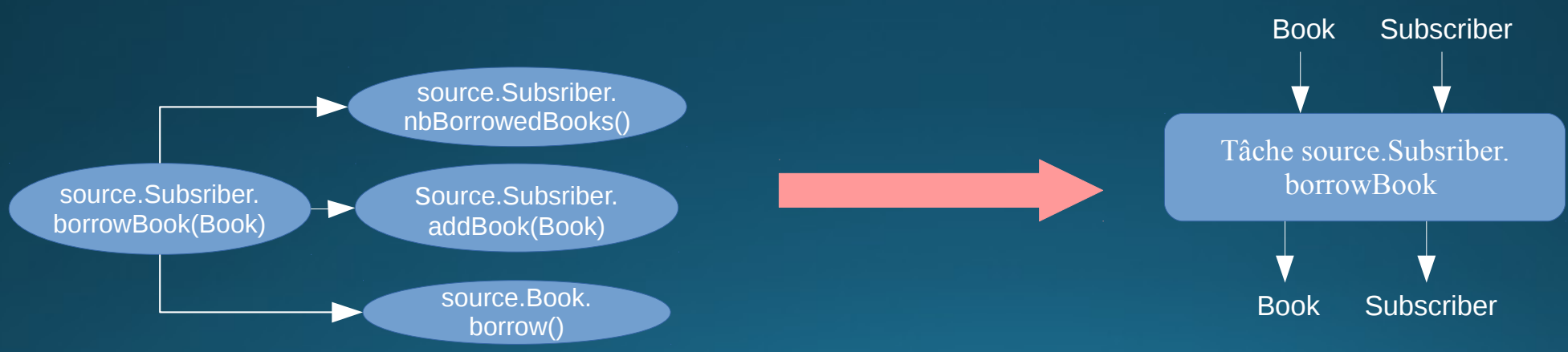


Analyse statique : identification des tâches (6/7)

- Identification des types des entrées d'une tâche
  - Types des paramètres de la méthode correspondante à cette tâche
  - Type de l'objet receveur de l'invocation de cette méthode
  - Types des variables globales (exemple : attribut statique publique dans java)

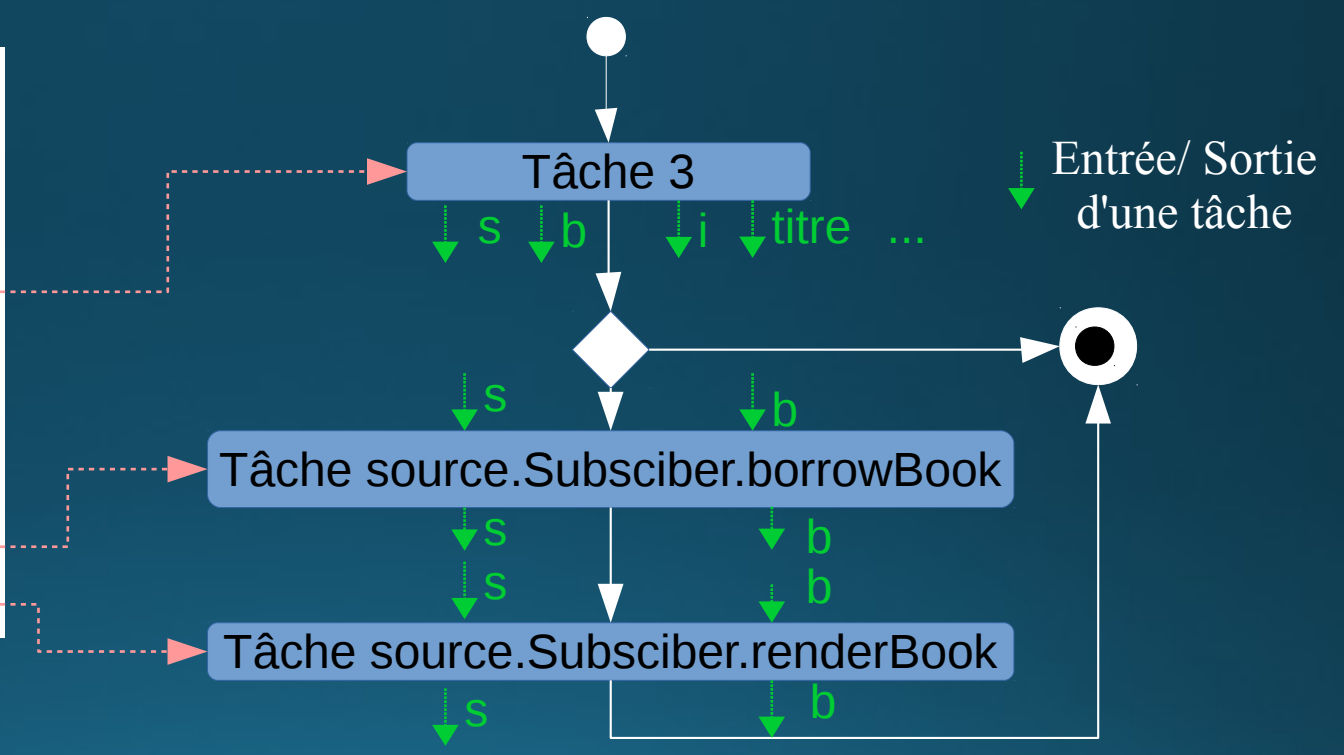
Analyse statique : identification des tâches (7/7)

- Identification des types des sorties d'une tâche
  - Types de la valeur retournée par la méthode correspondante à cette tâche
  - Types des entrées modifiées



Analyse statique : Extraction du flot de contrôle

```
public static void main(String[] args) {
  int i=0;
  Book b= new Book();
  while (i<5){
     Scanner input= new Scanner(System.in);
     String title= input.nextLine();
     String author= input.nextLine();
     b= new Book(title, author);
     i= i+1;
     LibBooks.add(b);}
  Subscriber s= new Subscriber(1);
  if (b.available){
     s.borrowBook(b);
     s.renderBook(b);}}}
```



Graphe de flot de contrôle

Analyse statique : Extraction d'un flot de données

- Forte probabilité que deux tâches sont dépendantes si :
  - Sortie de la première tâche possède le même nom et type de l'entrée de la deuxième

- Travail futur :
  - Proposition d'une solution pour la connexion des entrées et des sorties des tâches

Plan

- Contexte
  - Re-ingénierie d'applications orientées objets
  - Workflows
- Objectif
  - Extraction d'un workflow à partir d'une application orientée objet
- Une approche basée sur l'analyse statique
- Travaux connexes
- Conclusion et perspectives

Travaux connexes

- Rétro-ingénierie des applications orientée objets
- Génération de différents modèles:
  - Vue structurelle
    - Diagramme de classes
    - …
  - Vue temporelle
    - Workflow
    - Diagramme d'activité
    - Diagramme d'interaction
    - ...

Extraction des workflows (1/2)

- ***Model-driven business process recovery***. Zou, Y., Lau, T. C., Kontogiannis, K., Tong, T., and McKegney, R. (2004, November). In Reverse Engineering, 2004. Proceedings. 11th Working Conference on (pp. 224-233). IEEE.
  - ***Contribution :*** Approche de récupération d'un workflow à partir d'une application e-commerce
  - ***Objectif :*** Détermination des blocs de code qui implémente chaque tâche du processus métier fourni par la documentation
  - ***Discussion***
    - Correspondant d'une tâche dans le code source est un fragment de code représentant une logique métier

Extraction des workflows (2/2)

- ***An approach for extracting workflows from e-commerce applications***. Zou, Y., and Hung, M. (2006, June). In 14th IEEE International Conference on Program Comprehension (ICPC'06) (pp. 127-136). IEEE.
  - ***Contribution :*** Approche de récupération d'un workflow à partir d'une application e-commerce
  - ***Objectif :*** Détermination des blocs de code qui implémente chaque tâche du processus métier fourni par la documentation
  - ***Discussion***
    - Augmentation de l'abstraction du flot de contrôle généré pour qu'il soit proche de celui de la documentation

Extraction des diagrammes d'activités (1/2)

- ***Flowgen: Flowchart-based documentation for C++ codes***. «Kosower, D. A., and Lopez-Villarejo, J. J. (2015). Computer Physics Communications,196, 497-505 »
  - ***Contribution :*** Proposition d'un outil pour la récupération d'un ensemble de diagrammes d'activités interconnectés à partir du code C++ annoté
  - ***Objectif :*** Amélioration de la collaboration entre les développeurs et les scientifiques qui proposent les algorithmes implémentés
  - ***Discussion***
    - Intervention humaine pour l'ajout des annotations
    - Flot de données non récupéré

Extraction des diagrammes d'activités (2/2)

- ***CPP2XMI: Reverse Engineering of UML Class, Sequence, and Activity Diagrams from C++ Source Code.*** Korshunova, E., Petkovic, M., gj Van Den Brand, M., & Mousavi, M. R. (2006, October). In 2006 13th Working Conference on Reverse Engineering (pp. 297-298). IEEE.
  - ***Contribution :*** Proposition d'un outil pour la récupération du diagramme d'activité, de séquence et de classe à partir du code C++
  - ***Objectif :*** Vérification et validation des systèmes logiciels
  - ***Discussion***
    - ?

Extraction des diagrammes d'intéractions

- ***Reverse engineering of the interaction diagrams from C++ code.*** Tonella, P., & Potrich, A. (2003, September). . In *Software Maintenance, 2003. ICSM 2003. Proceedings. International Conference on* (pp. 159-168). IEEE

- ***Combining static and dynamic analyses to reverse-engineer scenario diagrams***.Labiche, Y., Kolbah, B., & Mehrfard, H. (2013, September). In *Software Maintenance (ICSM), 2013 29th IEEE International Conference on* (pp. 130-139). IEEE.

- ***Objectif :***
  - Compréhension des applications orientées objets

Critères de comparaison

- Source : code source (OO, procédural, etc)
- Cible : modèle généré (workflow, diagramme d'activité, etc)
- Analyse statique ou dynamique
- Extraction d'un flot de données
- Objectif

Tableaux comparatif

	Source	Cible	Analyse statique/ dynamique	Flot de données	Objective
Model driven business process recovery	Application e-commerce	workflow	Statique	✓	Tracability entre le code et la documentation
An approach for extracting workflows from e-commerce applications	Application e-commerce	workflow	Statique	✓	
CPP2XMI: Reverse Engineering of UML Class, Sequence, and Activity Diagrams from C++ Source Code	Application OO (C++)	Digrammes de classes, de séquences et d'activités	?	?	Validation et vérification
Flowgen : Flowchart-based documentation for C++	Application C++	Diagramme d'activité	Statique	✗	Amélioration de la collaborations entre les développeuses et les scientifique
Reverse engineering of interaction digram from C++	Application OO (C++)	Diagrammes d'interactions	Statique	✗	Compréhension d'un programme
Combining static and dynamic analysis to reverse-engineer scenario diagram	Application OO	Diagramme de séquence	Combinaison des deux	✗	Compréhension d'un programme

Plan

- Contexte
  - Re-ingénierie d'applications orientées objets
  - Workflows
- Objectif
  - Extraction d'un workflow à partir d'une application orientée objets
- Une approche basée sur l'analyse statique
- Travaux connexes
- Conclusion et perspectives

Conclusion

- Extraction d'un workflow à partir d'une application orientée objets en utilisant l'analyse statique du code.
- Modèle de correspondance entre les concepts d'une application OO et les concepts des workflows.
- Analyse statique pour :
  - Identifier des tâches
  - Extraire un flot de contrôle
  - Extraire un flot de données

Perspectives

- Implémentation et expérimentations
- Études de cas plus réalistes et complexes
- Migration vers un workflow à base de services ou de composants

Questions ?