SMARTBUILDERS
SUBOO : PHASE CONCEPTION

Membres du groupe :

AMROUCHE Sara
BOUKHATA Nora
FENEK Ouarda
KADRI Djaffar
MAURICE Nathan
OUDALI Saliha
OUERK Sara Yasmine

Organisation:
-Réunion
- GitHub
-Whatsapp

Plan

- Découpage en composants
- Diagramme de composants
- Diagramme de structure interne
- Design patterns façade + factory
- Diagramme de séquences inter-composants
- Configuration de tests
- Exemple d’implémentation de composant bouchon

Découpage en composants

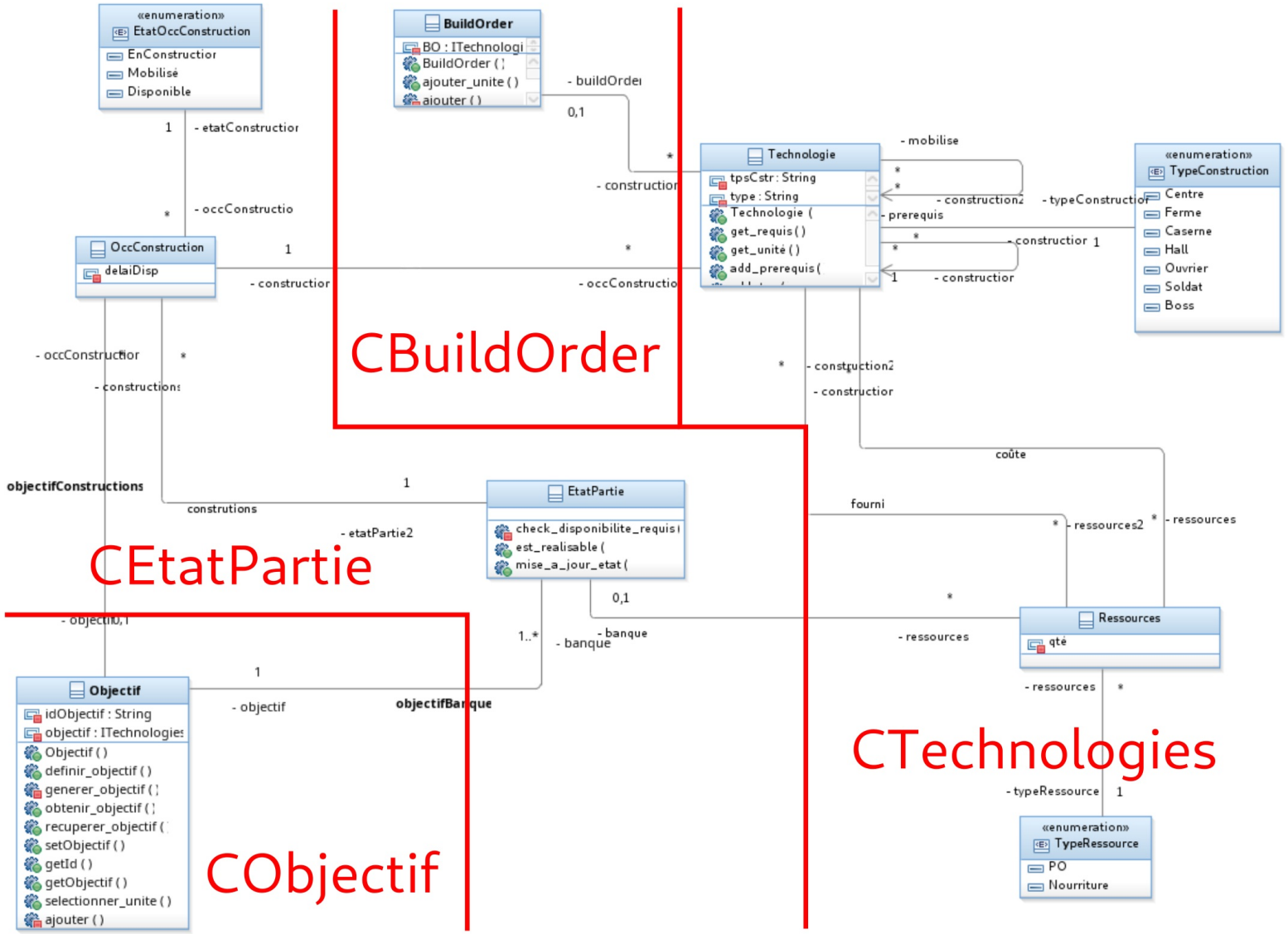


Diagramme de composants

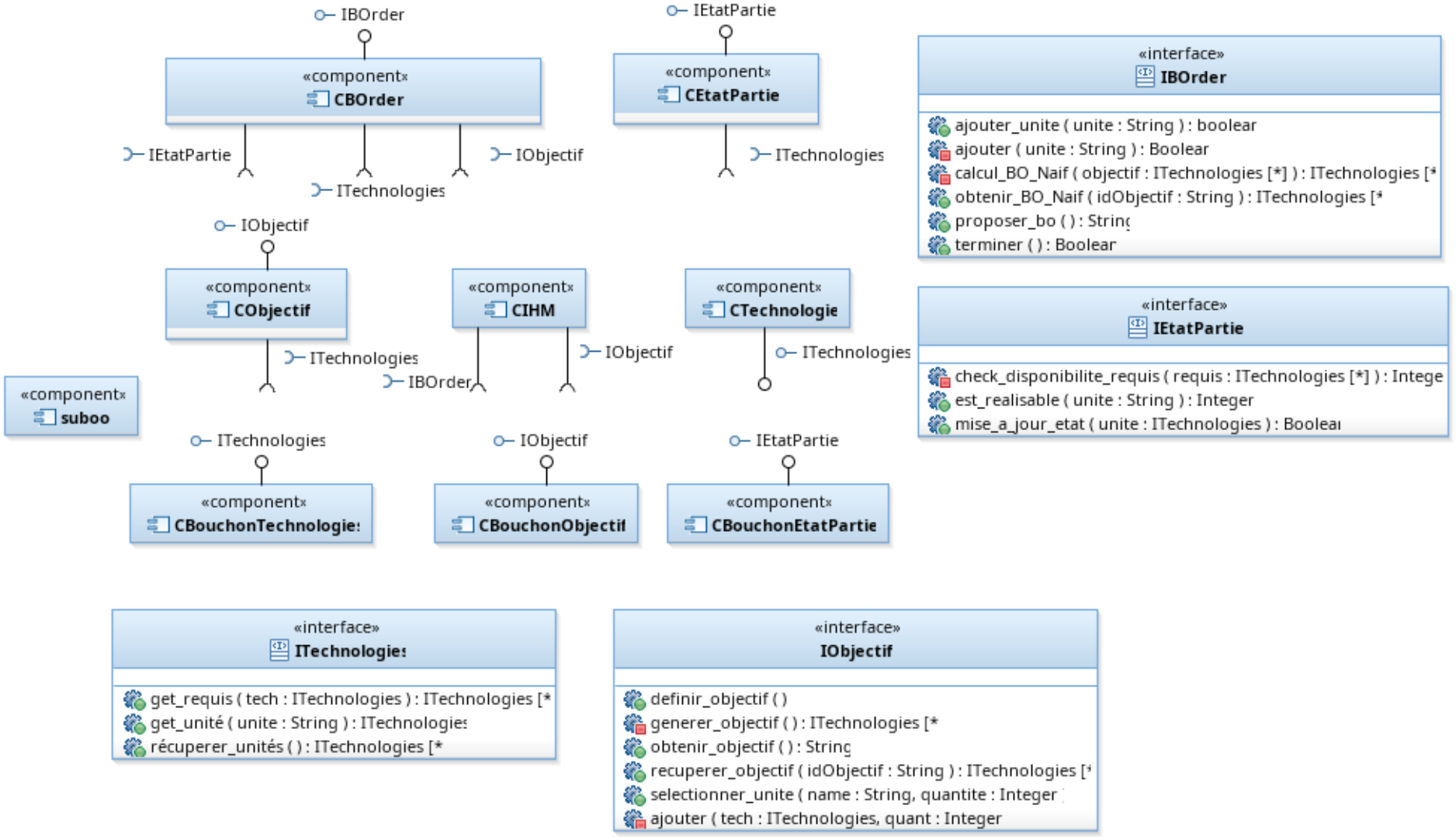
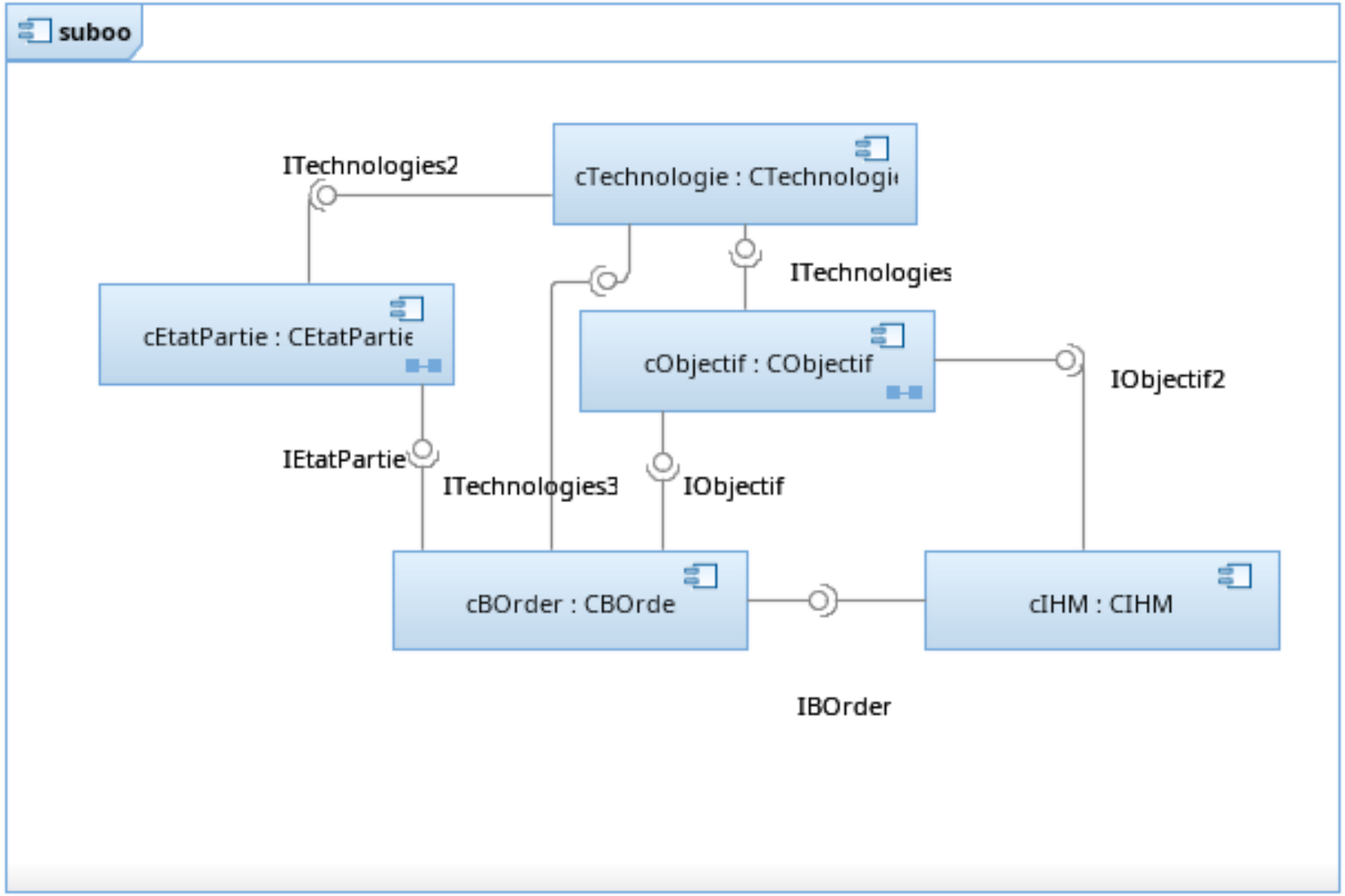
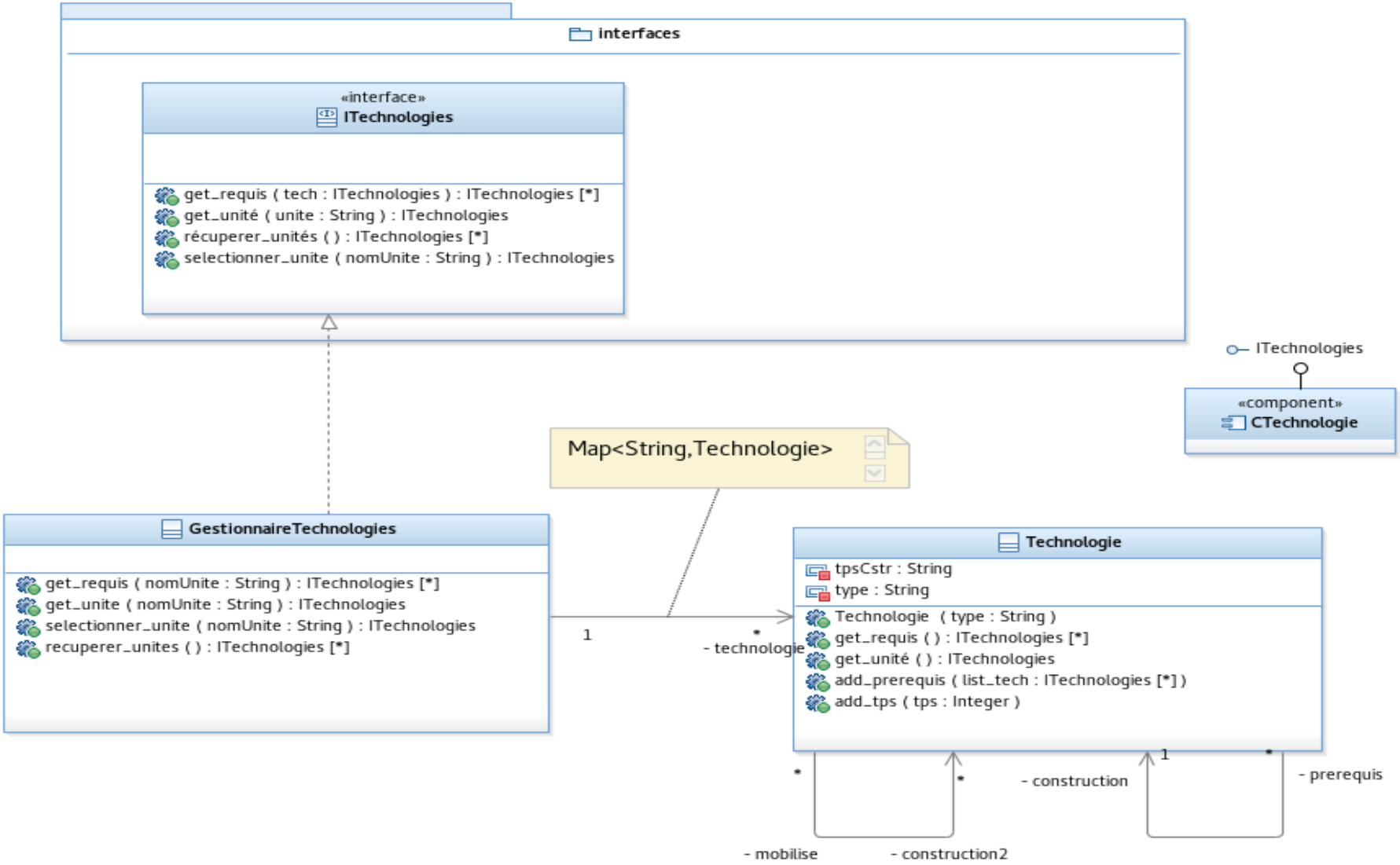


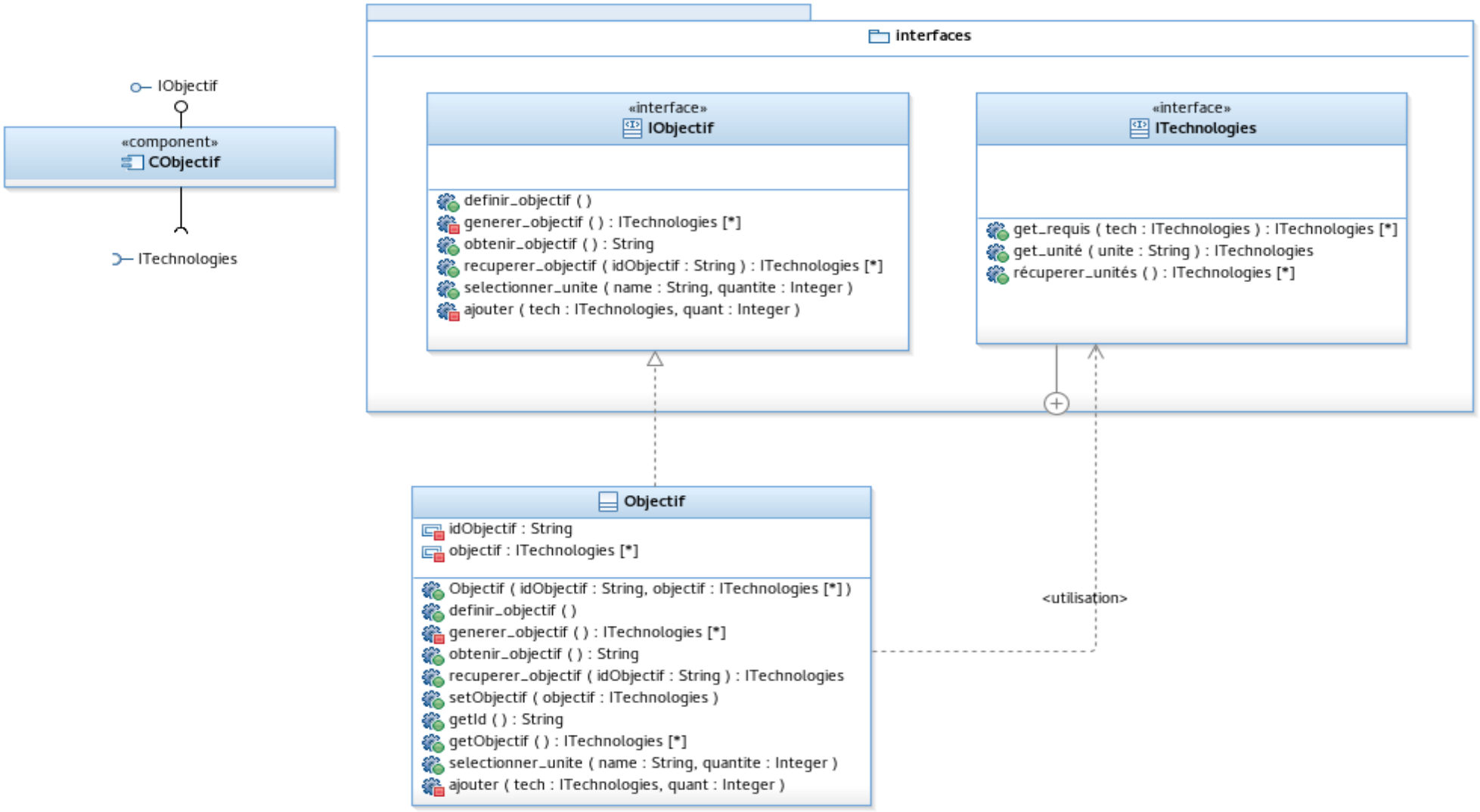
Diagramme de structure interne



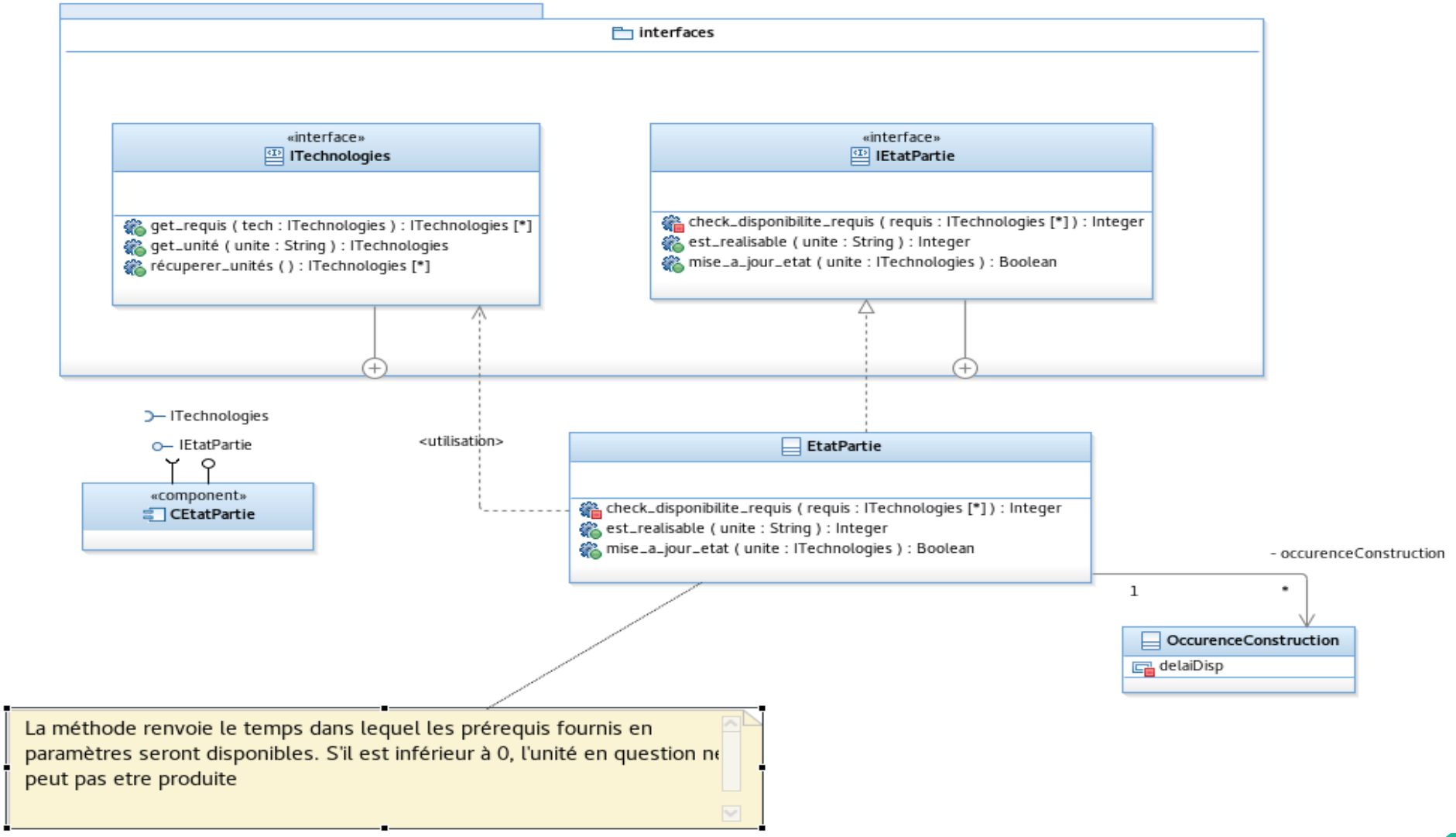
Design pattern façade
Technologie



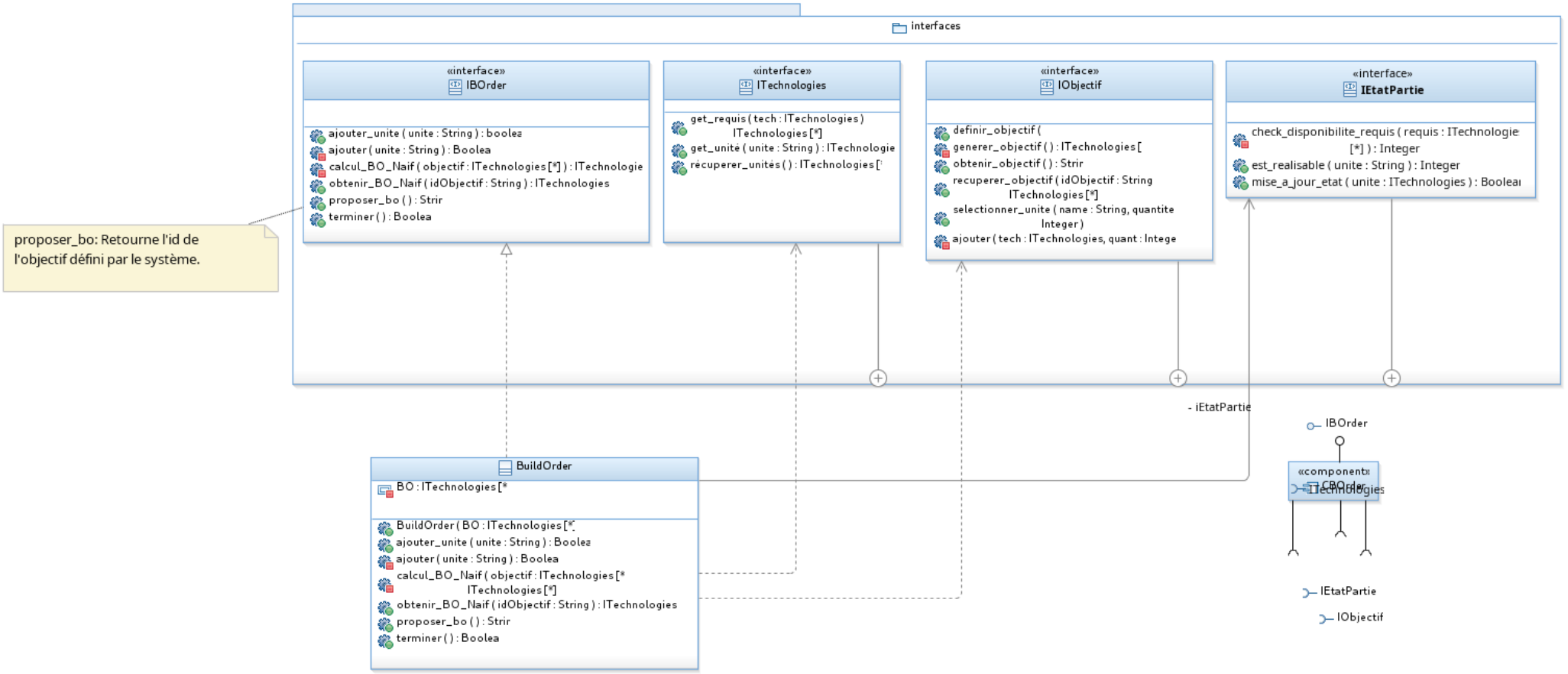
Design pattern façade
Objectif



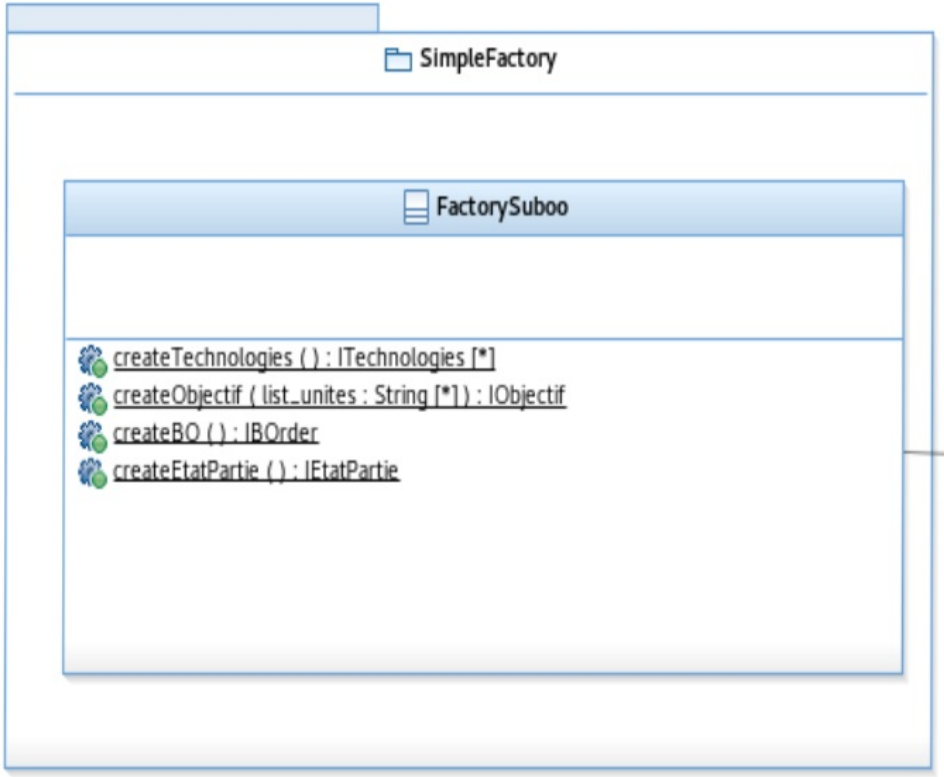
Design pattern façade
EtatPartie



Design pattern façade
BuildOrder



Design Pattern Factory




```
public static ArrayList<ITechnologies> createTechnologies(){

        return new GestionnaireTechnologie();

}

public static IObjectif createObjectif (ArrayList<String> list_unites){

        ArrayList<ITechnologies> res;

        for (String s : list_unites){

                res.add(new Technologie(s);

        }

        return new Objectif(res);

}
```

Diagramme de séquences inter-composants: Proposer BO

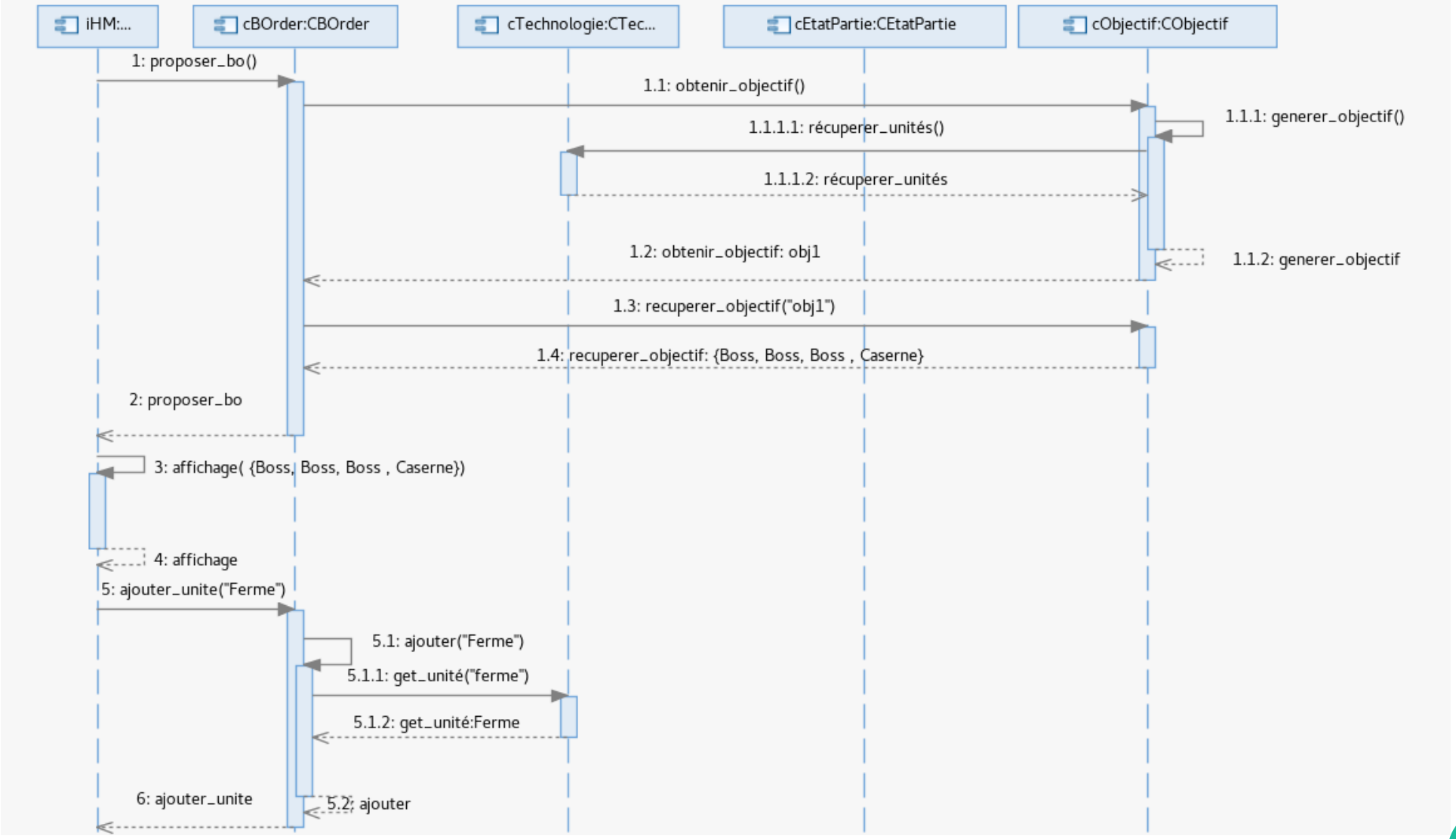


Diagramme de séquences inter-composants: Proposer BO

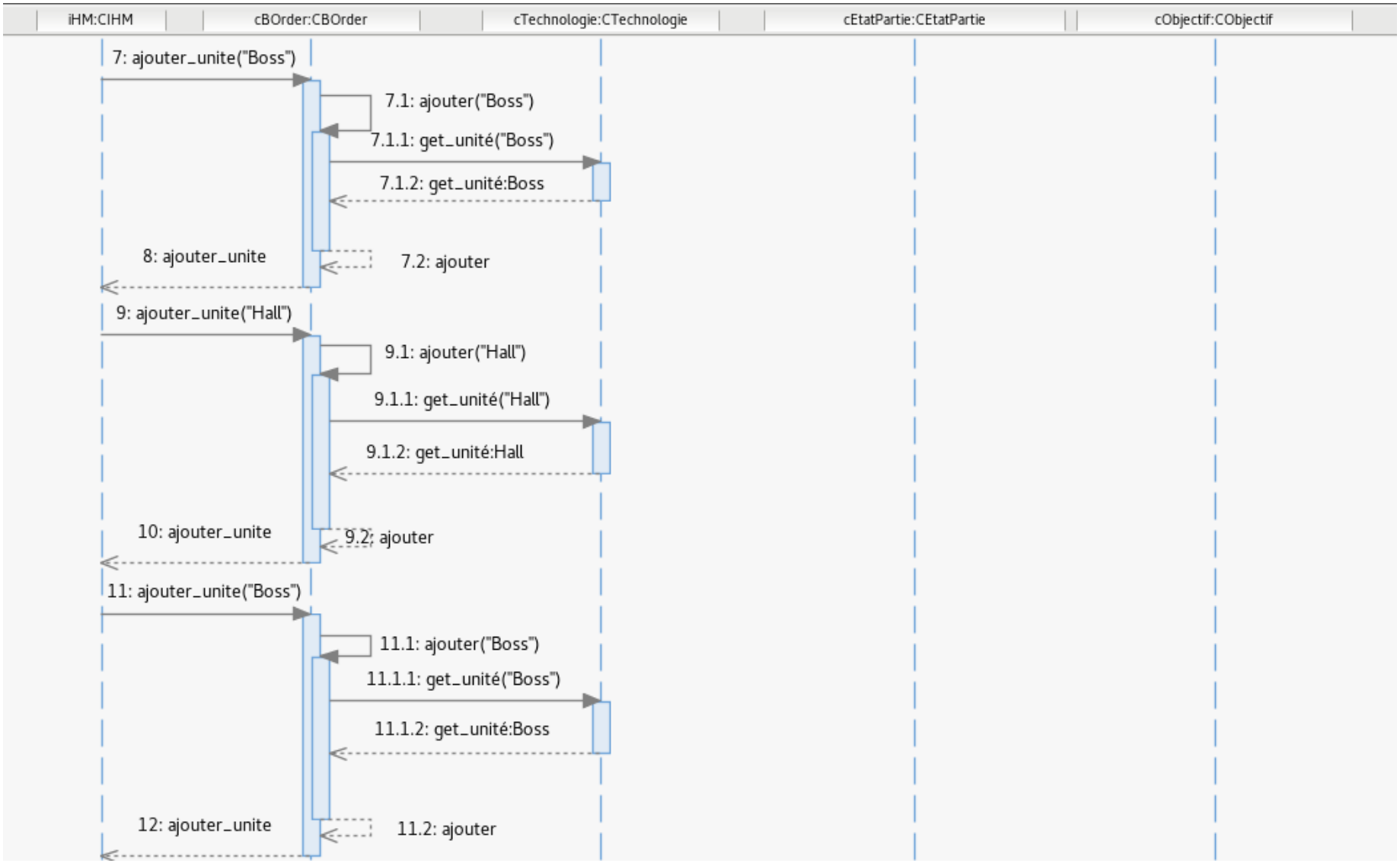


Diagramme de séquences inter-composants: Proposer BO

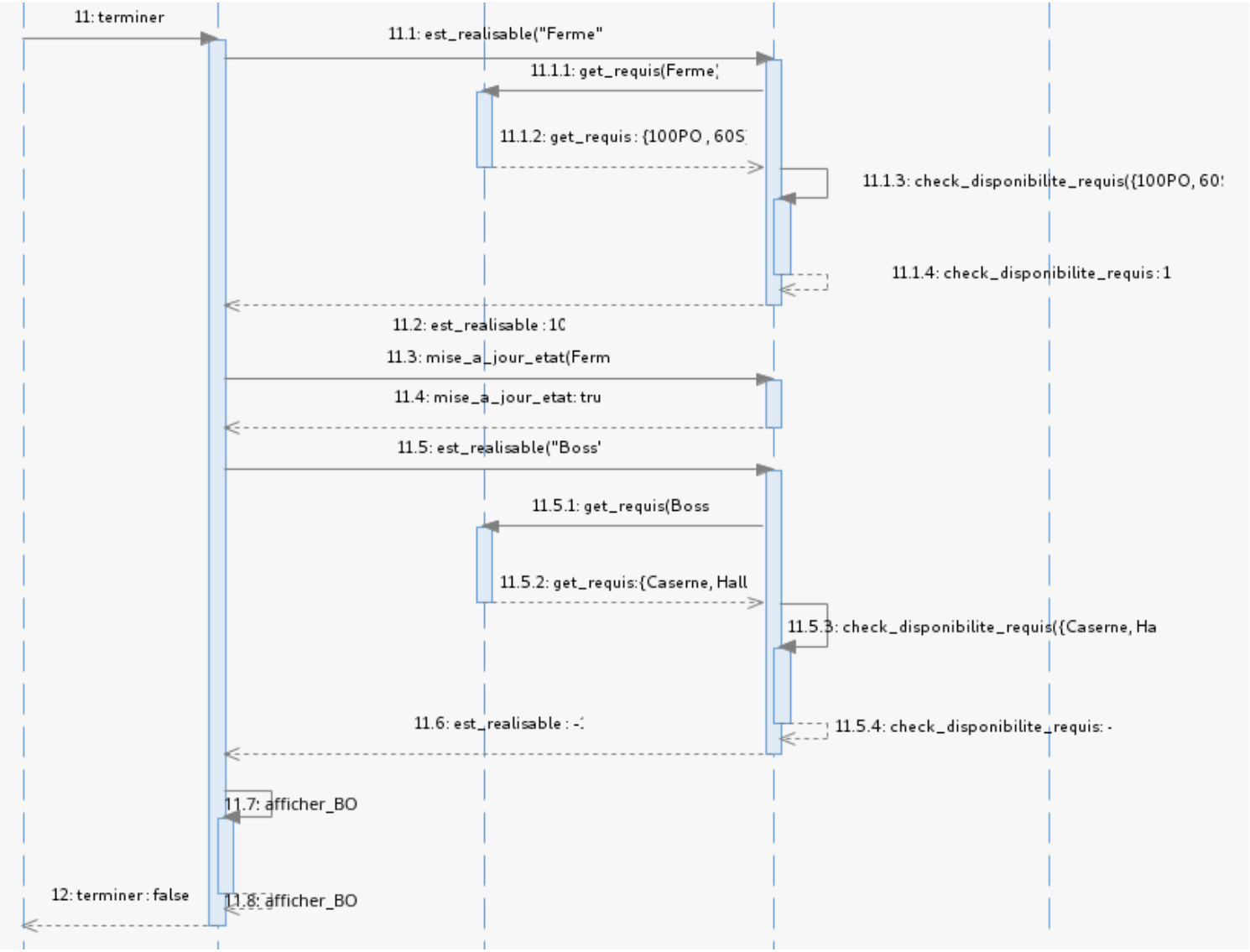


Diagramme de séquences inter-composants: Obtenir BO naïf

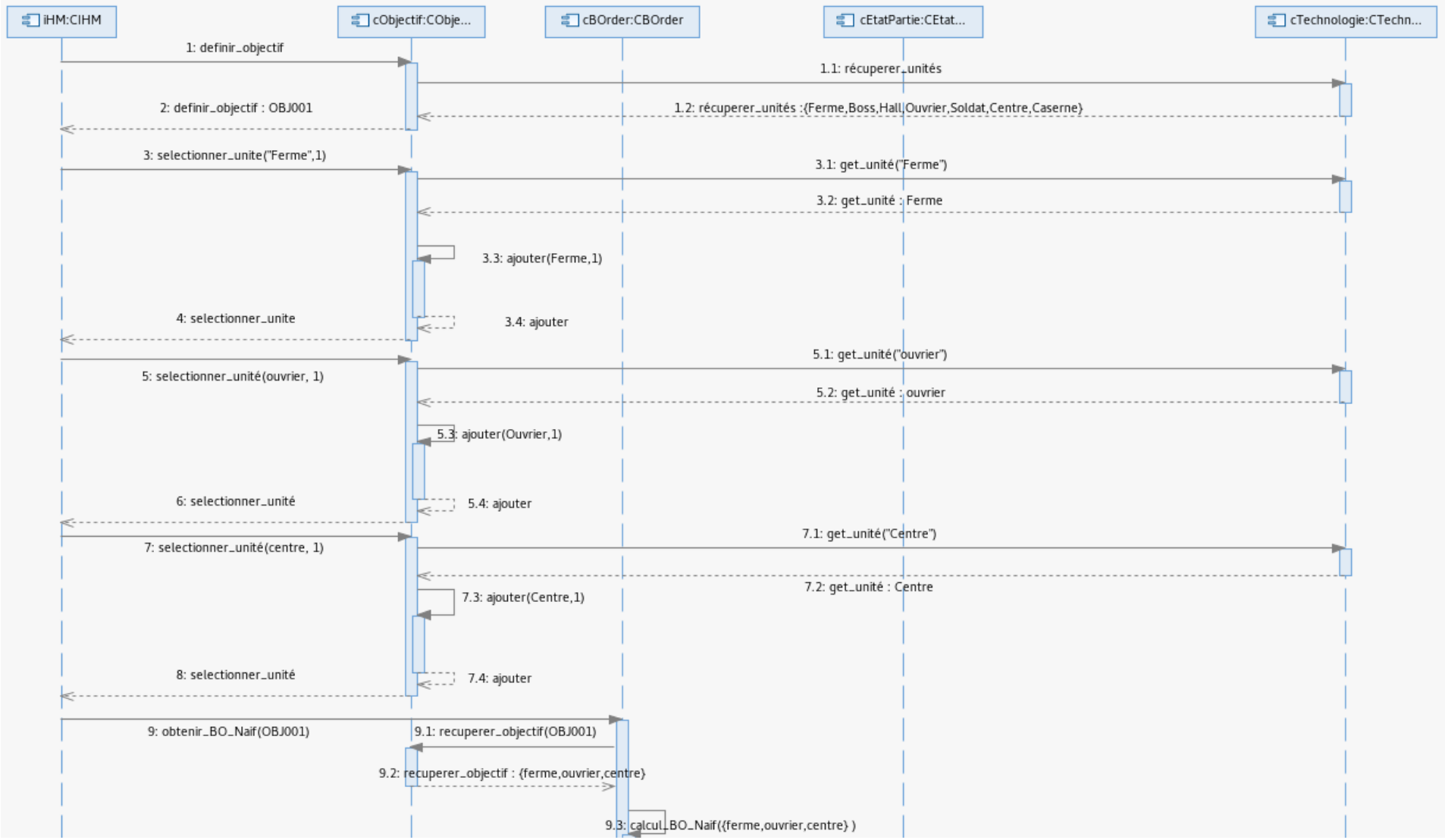
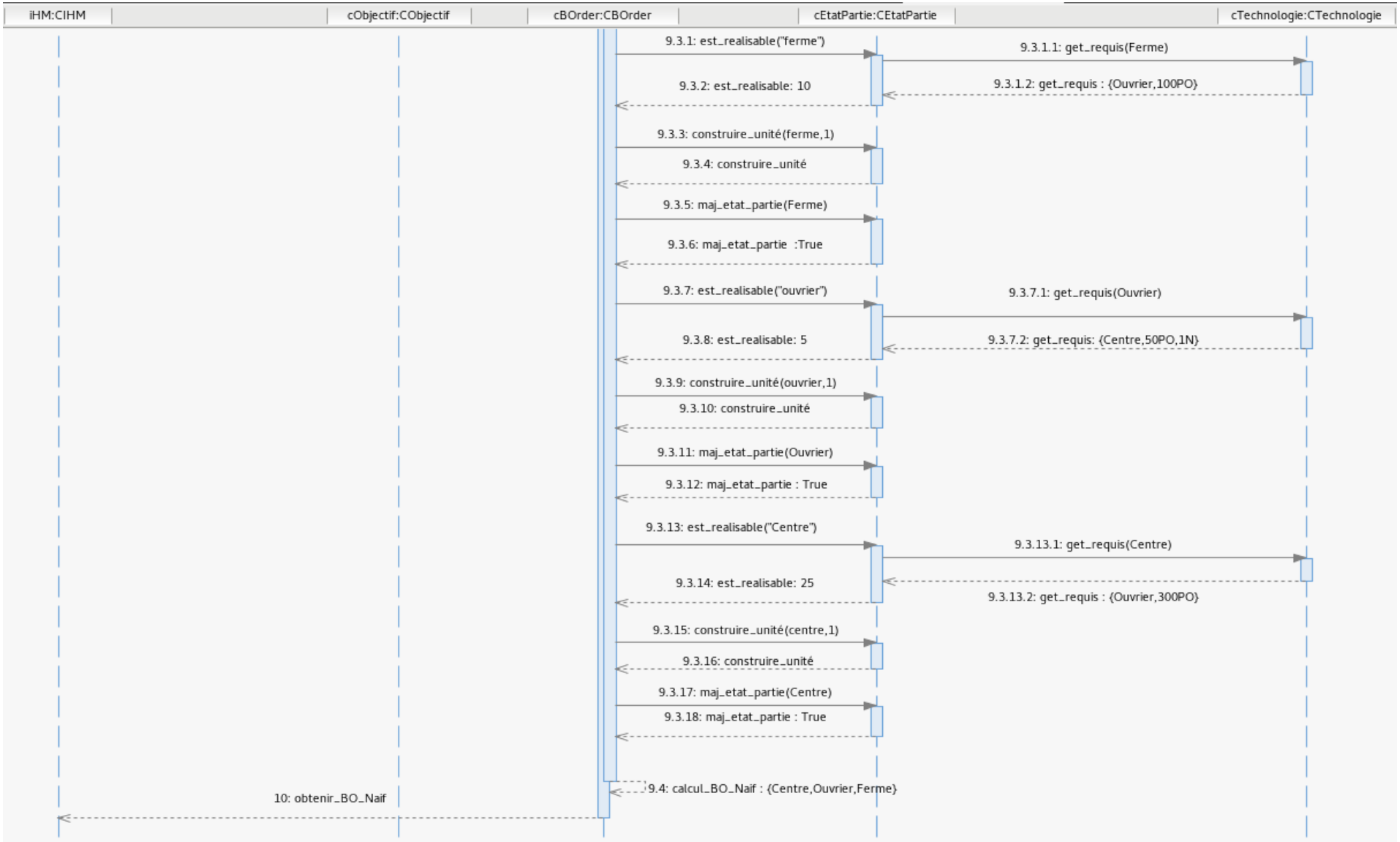
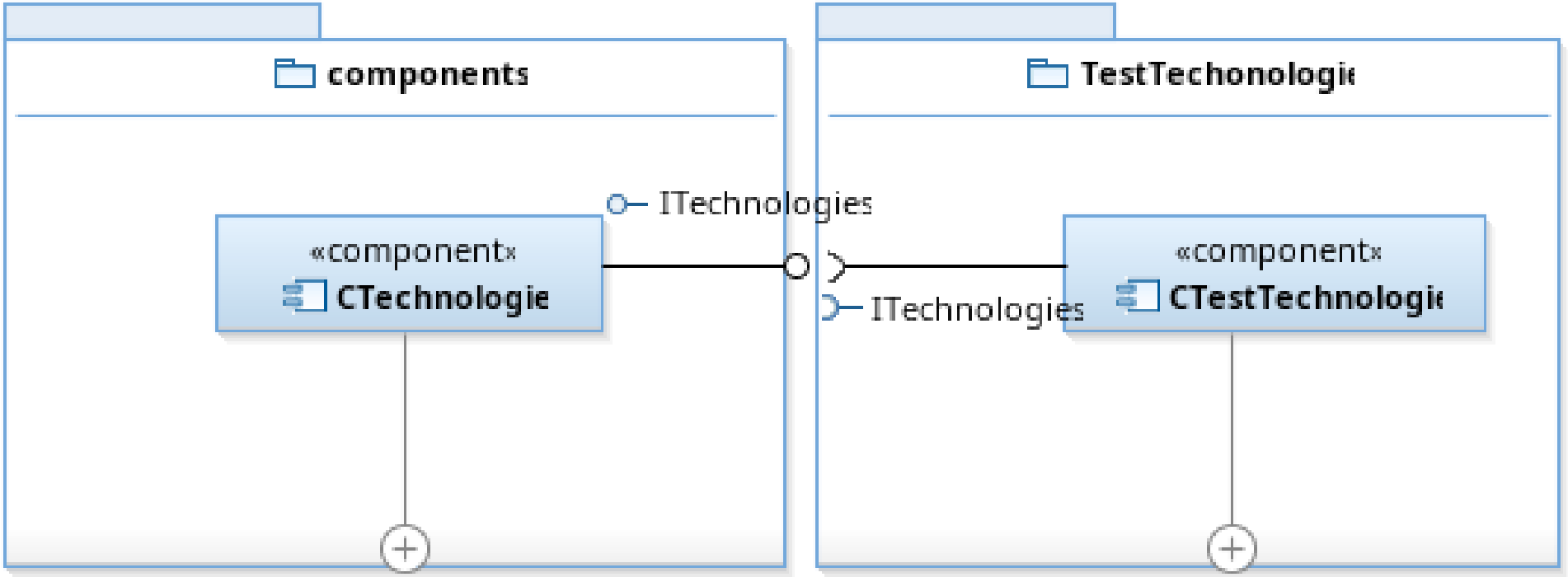


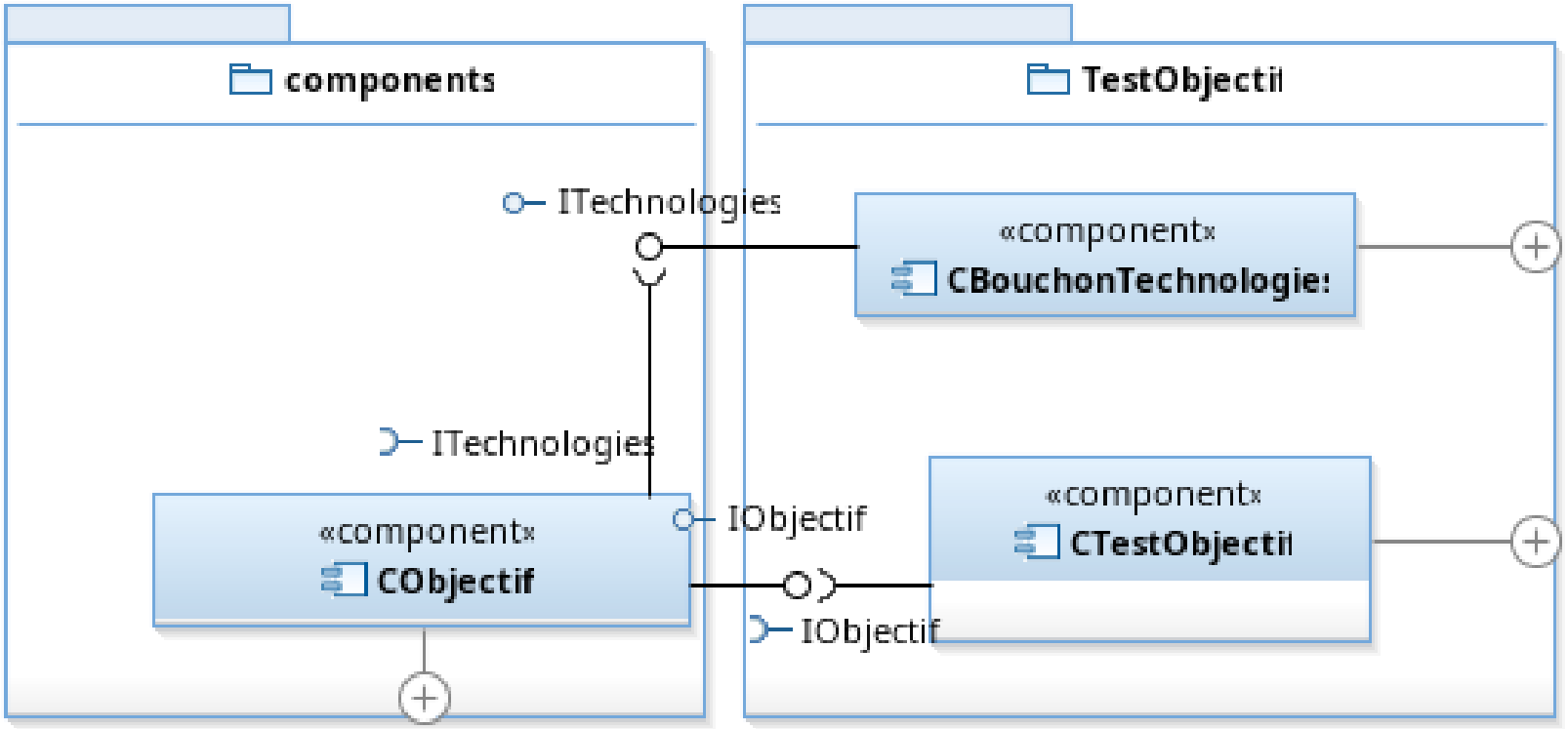
Diagramme de séquences inter-composants: Obtenir BO naïf



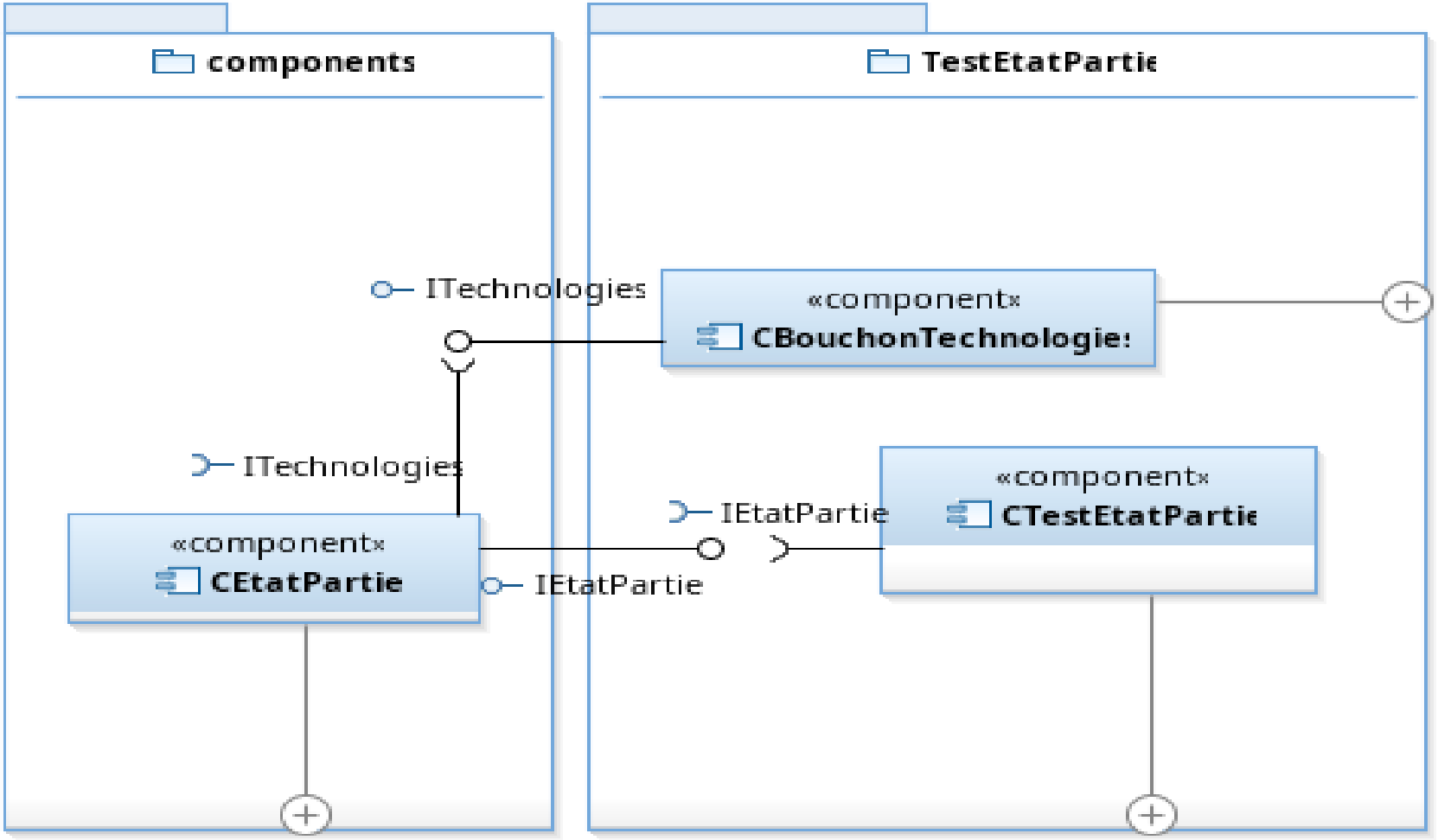
Configuration de tests
Test Technologie



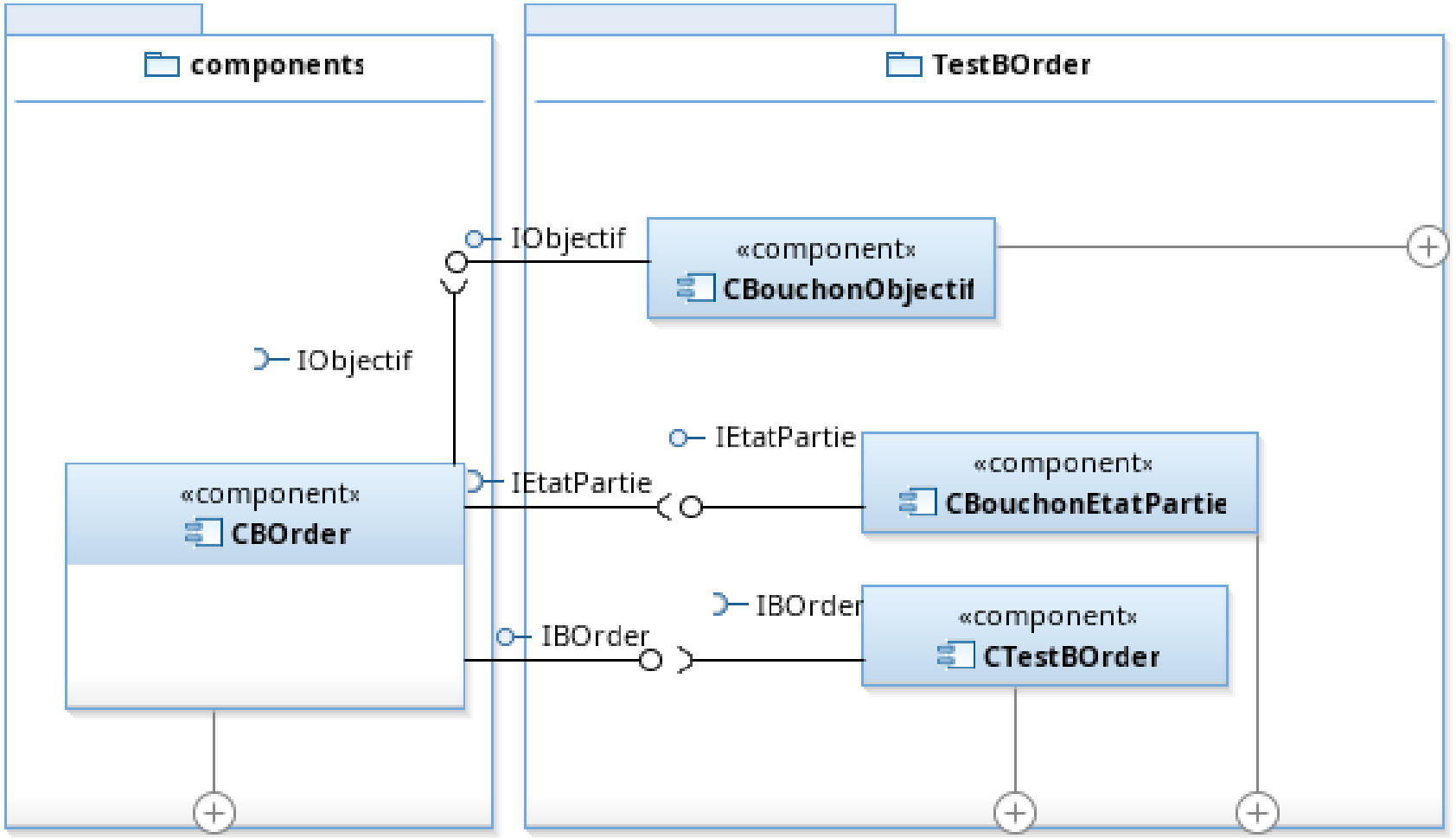
Configuration de tests
Test Objectif



Configuration de tests
Test État Partie



Configuration de tests
Test Build Order



implémentation de composant bouchon

Bouchon Technologie

```
package Test;

import java.util.ArrayList;


public class BouchonTechnologie {

    public ITechnologies get_unite(String unite){

        if (unite.equals("Boss")) return new Technologie("Boss");
        if (unite.equals("Ferme")) return new Technologie("Ferme");
        if (unite.equals("Ouvrier")) return new Technologie("Ouvrier");
        if (unite.equals("Caserne")) return new Technologie("Caserne");
        if (unite.equals("Centre")) return new Technologie("Centre");
        return null;

    }

    public ArrayList<ITechnologies> recuperer_unites(){
        ArrayList<ITechnologies> res = new ArrayList<ITechnologies>();
        res.add(new Technologie ("Boss"));
        res.add(new Technologie ("Ferme"));
        res.add(new Technologie ("Centre"));
        res.add(new Technologie ("Caserne"));
        res.add(new Technologie ("Hall"));
        res.add(new Technologie ("Ouvrier"));
        res.add(new Technologie ("Soldat"));
        res.add(new Technologie ("PO"));
        res.add(new Technologie ("N"));
        return res;
    }


}
```

implémentation de composant bouchon

Bouchon Etat

```java
package Test;

public class BouchonEtat {

    public int estRealisable(String unite){
        if ("ferme".equals(unite)) return 10;
        if ("Ouvrier".equals(unite)) return 20;
        if ("Centre".equals(unite)) return 20;
        if ("Boss".equals(unite)) return -1;
        return -1;
    }

}
```

Bouchon Objectif

```java
package Test;
import interfaces.ITechnologies;

public class BouchonObjectif {

    public String definir_objectif(){
        return "OBJ001";
    }

    public ArrayList<ITechnologies> recuperer_objectif(String id){

        ArrayList<ITechnologies> res = new ArrayList<ITechnologies>();
        res.add(new Technologie("Boss"));
        res.add(new Technologie("Boss"));
        res.add(new Technologie("Boss"));
        res.add(new Technologie("Caserne"));

        return res;
    }


}
```

Conclusion

Merci pour votre attention