Programmation avec android cours 4 (services)

Hadjila Fethallah

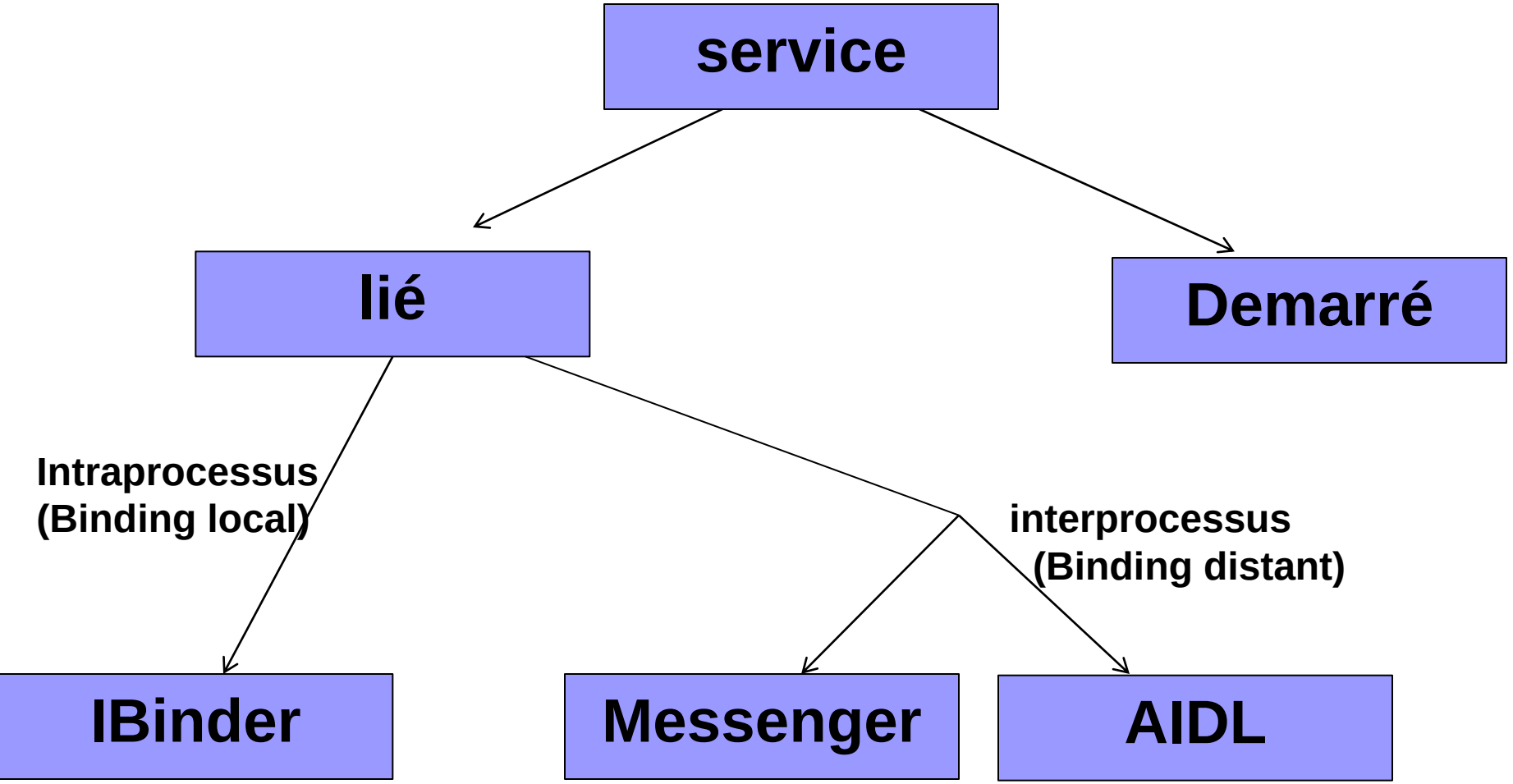
Maître de Conférences au Département d'Informatique

F_hadjila@mail.univ-tlemcen.dz

service

- Composant d'une application exécutant du code sans UI (généralement les longues tâches)
  - Exemples(telechargement, lecture de fichiers audio, requêtes BDD…)
- Le service continue l'execution même après suspension/destruction du composant appelant.
- Peut s'exécuter durant une période de temps indéfinie
- Par defaut un service s'execute dans le main thread du processus hote
- Un service est tué par le système si :
- pénurie de ressources (i.e pas assez de mémoire)
  - La susceptibilité d'être tué dépend de sa priorité
  - La priorité d'un service dépend (par défaut) de l'application qui l'utilise

service


service
lié
Demarré
Intraprocessus
(Binding local)
interprocessus
(Binding distant)
IBinder
Messenger
AIDL

Implementation d’un service

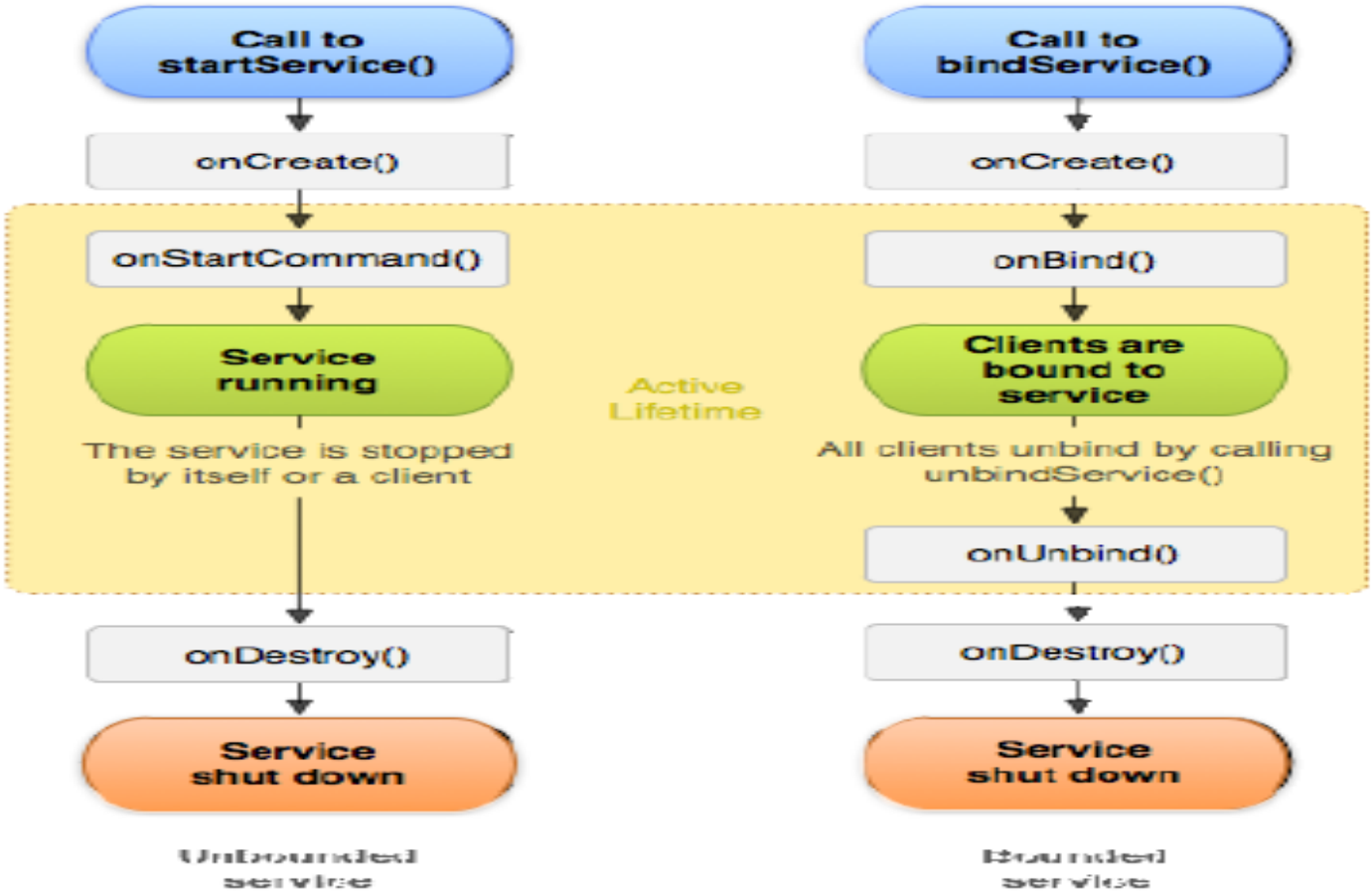
- Hériter de la classe Service / IntentService
- Ajouter le composant «service» dans le fichier manifest :
- <manifest ... >

```
  ...
  <application ... >
      <service android:name=".ExampleService" />
      ...
  </application>
</manifest>
```

Types de services

- **Service démarré (started service)**
  - Services simples (sans interaction avec le composant appelant), et ne retournant pas de résultât (en general).
  - S'executant de manière indéfinie
  - Lancés par un composant (ex: une activité) grâce à startService()
  - Appel de onStartCommand(Intent, flags, startId) pour l'execution de la tache
- **Service lié (*bound* service)**
  - Le composant client appelle bindService() qui lancera onBind()/onRebind()
  - Il permet l'échange de requêtes/réponses avec une application cliente (interaction complexe)
  - Il permet de réaliser les RPC (au sein du même système android)
  - Lorsque tous les clients deconnectent (unbind), le systeme detruit le service. Pas besoin d'arret explicite du service.
  - Deux classes candidates: messanger/ Interface AIDL

Cycle de vie d'un service



- Le lancement d'un service se fait avec un intent explicite

Service demarré

- On demande une réaction à un Intent en appelant Context.startService(Intent)
- La communication est uni-directionnelle (one way service)
- Le service est créé si nécessaire
- Le service peut être stoppé, par lui-même stopSelf, par celui qui l'a lancé stopService(Intent)
- Deux classes possibles: Service/ IntentService
- la methode onBind() retournera null.
- **Exemple:**
- **Intent intent = new Intent (this, hreadedDownloadService.class));**
- **intent.putExtra("URL", imageUrl); startService(intent);**

- **public class DownloadService extends Service { public int onStartCommand (Intent intent, int flags, int startId) { ... } }**

Service demarré avec intentServ

- IntentService: classe spécialisée pour les traitements long
- Il suffit d'implementer
- un constructeur de la classe IntentService
- pareil pour onHandleIntent().
- Le service est arrété automatiquement par android lorsque onHandleIntent() s'acheve.

IntentService

- public class HelloIntentService extends IntentService {
  public HelloIntentService() {
  super("HelloIntentService"); }
  /* le traitement des requetes est sequentiel dans le thread worker */
  protected void onHandleIntent(Intent intent) {
  // faire le travail ici (ex: telechargement).
  }
- public int onStartCommand(Intent intent, int flags, int startId) {
  Toast.makeText(this, "service starting", Toast.LENGTH_SHORT).show();
  return super.onStartCommand(intent,flags,startId); } }

Service demarré avec la classe service

- Possibilité de créer des threads gérant simultanément plusieurs requêtes (intents)
- La fonction onStartCommand() retourne 03 types de constantes
- START_NOT_STICKY
- START_STICKY
- START_REDELIVER_INTENT
- Un service continue l'execution meme apres l'achevement de onStartCommand()
- Pour arreter un service, il faut appeler explicitement stopSelf() ou stopService(),

Exemple détaillé(service démarré)

- Voir le TP5

Service lié

- Lorsqu'un composant (client) appelle Context.bindService(intent,serviceConnection,flags) pour se lier (bind) à un service

- flags:
  - BIND_AUTO_CREATE (démarre le service si nécessaire)
  - BIND_ADJUST_WITH_ACTIVITY(monte la priorité au même niveau que l'activité)
  - BIND_WAIVE_PRIORITY (pas de changement de priorité) => La méthode onBind()/onRebind() est appelée sur le service

Messenger

- Gère une queue de Messages inter-processus
- Traite toutes les requêtes dans un thread à part mais de manière séquentielle
- Messenger.getBinder() crée un IBinder
- Message.obtain() crée un Message
- Message.obtain(int what, int arg1, int arg2, Object obj)
- replyTo (optionel) Messenger pour la réponse
- Messenger.send(message) permet d'envoyer un message
- Un Handler permet de recevoir et traiter des Messages
- new Messenger(new Handler() { ... })

Workflow utilisant la classe Messenger (partie client)

- Implementer le listener ServiceConnection.
- Surcharger les methods:
  - onServiceConnected() : elle est appelée par le systeme pour deliver l'IBinder retourné par la methode onBind().
  - onServiceDisconnected() : elle est appelée par le systeme android lorsque la connection avec le service est perdue, (i.e le service est tué ou tombe en panne).
- Appeler bindService(), en passant l'implementation de ServiceConnection.
- Lorsque le systeme appelle onServiceConnected(), le client peut interagir avec le service en utilisant l'interface Ibinder
- Pour se deconnecter on appelle unbindService().

Workflow utilisant la classe Messenger (partie service)

- Le service implemente un Handler (sous forme de classe interne) qui traite les requetes du client.
- le service cree un objet Messenger qui possede une reference sur le Handler precedent.
- le Messenger cree un IBinder et le retourne au client (à partir de onBind()).
- Le client utilise le IBinder pour instancier un Messenger (possedant une reference sur le Handler du service), avec le messenger, le client peut envoyer des messages (requetes) au service.
- Le traitement des messages se fait dans la methode handleMessage() du Handler.

Service lié (Bound service)

- Les activitiés, les services, et les fournisseurs de contenu (content providers) peuvent se lier à un service (pas pour les broadcast receivers).
- Un service donné peut réaliser les deux types prédéfinis (started, bound).

Exemple détaillé(service lié)

- Voir le TP6

FIN du Cours4