使い方ですが，
- とりあえず，**ccnrun.batダブルクリックするとシミュレータが動きます**．全ノードが全ファイルをDLするとシミュレータは完了しますが，まだ完全ではなく，多分終わらないので，途中でCtrl+Cで終了させてください．そして，**ccn/ccnlog.csvにログが上書きモードで出力されます**．Excelで開くと，大体は分かると思います．
- docs/api/index.htmlで，javadocが見れます．が，コメントが不十分です．

1．キャッシュアルゴリズムについて：

- キャッシュアルゴリズムですが，src/net/gripps/ccn/caching/BaseCachingAlgorithm.javaのクラスを「継承」したクラスを作ることになります．今は，OnPathCaching.javaが，BaseCachingAlgorithmを継承して使われています．このOnPathCahing.javaのような継承の仕方をして，新規クラスを作ってください．
- **新規キャッシュアルゴリズムの開発方法：**ccn.propertiesの，「ccn_caching_no=0」→「ccn_caching_no=1」（使われるインデックス番号）とし，「ccn_caching_allnum=1」を「ccn_caching_allnum=2」（アルゴリズムの候補数）とします．そして，/src/net/gripps/ccn/core/CCNRouter.javaの

```
this.cachings[0] = new OnPathCaching();
this.cachings[1] = new NoCaching();
/***ここまで**/
this.usedCaching = this.cachings[CCNUtil.ccn_caching_no];
```

を

```
this.cachings[0] = new OnPathCaching();
this.cachings[1] = new NoCaching();
this.cachings[2] = new 新規クラス名();
/***ここまで**/
this.usedCaching = this.cachings[CCNUtil.ccn_caching_no];
```

としてください．これでOKかと思います．

2. FIBルーティングアルゴリズムについて:

- FIBルーティングアルゴリズムでは，/src/net/gripps/ccn/fibrouting/BaseRouting.javaのクラスを「継承」したクラスを作ることになります．今は，ChordDHTRouting.javaが，BaseRouting.javaを継承して使われています．ぶっちゃけこれを使って試験してもいいけど
- **新規FIBルーティングアルゴリズムの開発方法:**ccn.propertiesの，「ccn_routing_no=0」→「ccn_routing_no=1」(使われるインデックス番号)とし，「ccn_routing_allnum=1」を「ccn_routingg_allnum=2」(アルゴリズムの候補数)とします．そして，/src/net/gripps/ccn/process/CCNMgr.javaの，

```
/***ここに，ルーティングアルゴリズムを列挙してください***/
this.routings[0] = new ChordDHTRouting(this.nodeMap, this.routerMap);
/*************ここまで************************/
this.usedRouting = this.routings[CCNUtil.ccn_routing_no];
```

を

```
/***ここに，ルーティングアルゴリズムを列挙してください***/
this.routings[0] = new ChordDHTRouting(this.nodeMap, this.routerMap);
this.routings[1] = new 新しいクラス名(this.nodeMap, this.routerMap);
/*************ここまで************************/
this.usedRouting = this.routings[CCNUtil.ccn_routing_no];
```

としてください．

3. BreadCrumbs(パンくず)アルゴリズムについて:

・パンくずアルゴリズムですが，/src/net/gripps/ccn/breadcrumbs/BaseBreadCrumbsAlgorithm.javaのクラスを「継承」したクラスを作ることになります．今は，BreadCrumbsAlgorithm.javaが，BaseBreadCrumbsAlgorithm.javaを継承して使われています．

- **新規パンくずアルゴリズムの開発方法:**ccn.propertiesの，「ccn_bc_allnum=2」→「ccn_bc_allnum=3」(アルゴリズムの総数)とし，「ccn_bc_enable」の値を田例えば「2」(指定するアルゴリズムの番号(0から開始))とします．そして，/src/net/gripps/ccn/core/CCNRouter.javaの

```
this.bcs[0] = new NoBreadCrumbsAlgorithm();
this.bcs[1] = new BreadCrumbsAlgorithm();

this.usedBC = this.bcs[CCNUtil.ccn_bc_enable];
```

を

```
this.bcs[0] = new NoBreadCrumbsAlgorithm();
this.bcs[1] = new BreadCrumbsAlgorithm();
this.bcs[2] = new 新しいクラス名();

this.usedBC = this.bcs[CCNUtil.ccn_bc_enable];
```

としてください．

・コンパイルは，antを使います．build.xmlと同じディレクトリで，コマンドプロンプトで「ant build」を打つとコンパイルします．
antのインストール方法は，ネットで調べてみるのがいいです．
とりあえず環境変数にANT_HOME=antのディレクトリ（例：c:\ant)を設定し，pathに%ANT_HOME%\bin を設定する必要があります．

4．Churn（ルータ参加・離脱）アルゴリズムについて：

－ Churnアルゴリズムでは，/src/net/gripps/ccn/churn/BaseChurnResilienceAlgorithm.javaのクラスを「継承」したクラスを作ることになります．今は，ChordDHTCRAlgorithm.javaがBaseChurnResilienceAlgorithm.javaを継承して使われています．ぶっちゃけこれを使って試験してもいいけど
－ **新規Churnアルゴリズムの開発方法：**ccn.propertiesの，「ccn_churn_enable=0」→「ccn_churn_enable=2」（使われるインデックス番号）とし，「ccn_churn_allnum=2」を「ccn_routingg_allnum=3」（アルゴリズムの候補数）とします．そして，/src/net/gripps/ccn/process/CCNMgr.javaの，

```
//**********Churningアルゴリズム***************/
this.churns = new BaseChurnResilienceAlgorithm[CCNUtil.ccn_churn_allnum];
this.churns[0] = new NoChurnAlgorithm(this.usedRouting);
this.churns[1] = new ChordDHTCRAlgorithm((ChordDHTRouting)this.usedRouting);
this.usedChurn = this.churns[CCNUtil.ccn_churn_enable];
//************:ここまで************************/
```

を

```
//**********Churningアルゴリズム***************/
this.churns = new BaseChurnResilienceAlgorithm[CCNUtil.ccn_churn_allnum];
this.churns[0] = new NoChurnAlgorithm(this.usedRouting);
this.churns[1] = new ChordDHTCRAlgorithm((ChordDHTRouting)this.usedRouting);
this.churns[2]=new 新しいクラス名前();
this.usedChurn = this.churns[CCNUtil.ccn_churn_enable];
//************:ここまで************************/
```

としてください．

以上