流れ グラフの作成 →グラフのインスタンスの作成 →add nodeひつようか

ノードの作成の方が先か yamlから読み出して pureなノードのインスタンスを作成→tagに色々追加する この時点で許されている遷移がわかっている next nodeみたいな感じでidのリストを作成する

ノードが全て読み込めたら，　→graphの作成 ①graph instanceが作成される ②add node ??　これはすでにnodeがリストを持っている時点で不要なプロセスかも③ノードを全探索しながらget next?を呼び出して，そのidを見ながらedge?

これで状態遷移を管理するぐらふのさくせいがかんりょうした・

使われ方

@今の状態は何ノード？ → get current node(dict{str, str}) 引数は rosnodeからえた(lifecyclenode, state)のタプルの辞書か？いや普通に 辞書

component

切り替え判定の流れ

rosから外界を得る graphから現在のidを得る，その時次にいける状態も得る？？

例えば ある地点Pに十分近づいたらid=1→id2に遷移したい場合

ノード遷移ごとに遷移したい条件が異なる場合どうする？？

C++の実装で，　状態遷移をグラフで管理して，そのグラフをも持っているクラスで状態遷移の判定を行いたい，という状況を考えます．　判定するクラスをAとしておきましょう．Aは外から情報を得ることができます．これは条件判定に必要な変数としましょう．温度とか時刻とかをイメージしてください.

状態はグラフで管理しています．一つの状態が一つのノード，状態遷移がエッジとして表現されています．　Aはこのグラフを持っています（ノード自体はグラフが持っていて直接アクセスできません）グラフは，現在の状態がなんなのかをノードのidを通じて教える関数を持っています．　またnode idを引数に，そのノードに対応する状態や，そのノードが次に行けるノードのリストなどの情報を返すget node infoの様な関数を持っているとします.

graphの機能をつかうことでAは状態を管理します．また外界の情報を得ながら状態遷移の判定を行います.

ここで判定を実現するための実装の方法について議論したいです．　Aが持つ機能は外界から情報を得る関数，現在の状態のidをgraphから得る関数，　状態遷移を行うかどうかを判定する関数 →遷移すべきでない時は何も返さず，遷移する時に，遷移するための方法（これは　structで適当に定義されているとしましょう.）自体を返すものとします.

状態によって，遷移すべきかどうか，もしくはなんの状態に遷移すべきかの判定が変わるとします 例えばnode1,node2, node3があって 1→2，2→1 1→3，3→1 の遷移が可能だとします．しかし，　それぞれの遷移はことなるじょうけんにもとづいて行われるとします.

Aはこれを管理するためにまず現在の状態をグラフから得ます． 次に判定関数に現在の状態と，自分が持っている外界の情報を引数に入れます 関数の中で判定します． 結果をreturnします．

ここで，現在の状態に応じて判定のロジックを変えるということをたとえばcaseでやってもいいのですが，これは結構大変です． なんらかの方法で，判定条件自体はノードごとに外で実装してそとから使う，みたいなことを実現したいんですよね．